PROGRAMMING FOR PROBLEM SOLVING

UNIT 4 ONE SHOT

WELCOME

INTELLIGENCE LEARNING

What is an Array in C?

An **array** in C is a **collection of elements of the same data type** stored **at contiguous memory locations**. It allows you to store multiple values under a single variable name, making it easier to manage large amounts of related data.

Syntax:-

data_type array_name[size];

C program to multiply two matrices and display the result matrix

```
#include <stdio.h>

int main() {
    int a[10][10], b[10][10], result[10][10];
    int r1, c1, r2, c2, i, j, k;

    // Input rows and columns for first matrix
    printf("Enter rows and columns for first matrix: ");
    scanf("%d %d", &r1, &c1);

    // Input rows and columns for second matrix
    printf("Enter rows and columns for second matrix: ");
    scanf("%d %d", &r2, &c2);

    // Check if multiplication is possible
    if(c1 != r2) {
        printf("Matrix multiplication not possible!\n");
        return 0;
    }
```

```
// Input elements of first matrix
printf("Enter elements of first matrix:\n");
for(i = 0; i < r1; i++) {
   for(j = 0; j < c1; j++) {
      scanf("%d", &a[i][j]);
   }
}

// Input elements of second matrix
printf("Enter elements of second matrix:\n");
for(i = 0; i < r2; i++) {
   for(j = 0; j < c2; j++) {
      scanf("%d", &b[i][j]);
   }
}
```

```
// Multiply matrices
for(i = 0; i < r1; i++) {
    for(j = 0; j < c2; j++) {
        result[i][j] = 0;
        for(k = 0; k < c1; k++) {
            result[i][j] += a[i][k] * b[k][j];
        }
    }
}

// Display the result
printf("Resultant matrix is:\n");
for(i = 0; i < r1; i++) {
    for(j = 0; j < c2; j++) {
        printf("%d\t", result[i][j]);
    }
    printf("\n");
}

return 0;
}
```

C program to multiply two 3×3 matrices and store the result in another matrix.

```
#include <stdio.h>

int main() {
    int a[3][3], b[3][3], result[3][3];
    int i, j, k;

    // Input elements of first matrix
    printf("Enter elements of first
3x3 matrix:\n");
    for(i = 0; i < 3; i++) {
        for(j = 0; j < 3; j++) {
            scanf("%d", &a[i][j]);
        }
    }
```

```
// Input elements of second matrix
   printf("Enter elements of second 3x3
matrix:\n");
   for(i = 0; i < 3; i++) {
      for(j = 0; j < 3; j++) {
         scanf("%d", &b[i][j]);
      }
   }

   // Multiply matrices
   for(i = 0; i < 3; i++) {
      for(j = 0; j < 3; j++) {
         result[i][j] = 0;
         for(k = 0; k < 3; k++) {
            result[i][j] += a[i][k] * b[k][j];
         }
      }
   }
```

```
    // Display result matrix
    printf("Resultant matrix after
multiplication:\n");
    for(i = 0; i < 3; i++) {
        for(j = 0; j < 3; j++) {
            printf("%d\t", result[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

SAMPLE INPUT:-

Enter elements of first 3x3 matrix:
1 2 3
4 5 6
7 8 9

Enter elements of second 3x3 matrix:
9 8 7
6 5 4
3 2 1

OUTPUT:-

Resultant matrix after multiplication:

30	24	18
84	69	54
138	114	90

C program to add two 3×3 matrices and store the result in another matrix.

```
#include <stdio.h>

int main() {
    int a[3][3], b[3][3], sum[3][3];
    int i, j;

    // Input elements of first matrix
    printf("Enter elements of first 3x3 matrix:\n");
    for(i = 0; i < 3; i++) {
        for(j = 0; j < 3; j++) {
            scanf("%d", &a[i][j]);
        }
    }

    // Input elements of second matrix
    printf("Enter elements of second 3x3 matrix:\n");
    for(i = 0; i < 3; i++) {
        for(j = 0; j < 3; j++) {
            scanf("%d", &b[i][j]);
        }
    }
```

```
    // Add matrices
    for(i = 0; i < 3; i++) {
        for(j = 0; j < 3; j++) {
            sum[i][j] = a[i][j] + b[i][j];
        }
    }

    // Display result matrix
    printf("Sum of the two matrices is:\n");
    for(i = 0; i < 3; i++) {
        for(j = 0; j < 3; j++) {
            printf("%d\t", sum[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

SAMPLE INPUT:-

Enter elements of first 3x3 matrix:
1 2 3
4 5 6
7 8 9

Enter elements of second 3x3 matrix:
9 8 7
6 5 4
3 2 1

OUTPUT:-

Sum of the two matrices is:

10	10	10
10	10	10
10	10	10

C Program: Find Largest Number in 4×4 Matrix

```
#include <stdio.h>

int main() {
    int matrix[4][4];
    int i, j, max;

    // Input elements of 4x4 matrix
    printf("Enter elements of 4x4
matrix:\n");
    for(i = 0; i < 4; i++) {
        for(j = 0; j < 4; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }

    // Initialize max with the first element
    max = matrix[0][0];
```

```
// Find the largest element
   for(i = 0; i < 4; i++) {
      for(j = 0; j < 4; j++) {
         if(matrix[i][j] > max) {
            max = matrix[i][j];
         }
      }
   }

   // Display the largest element
   printf("The largest number in the matrix
is: %d\n", max);

   return 0;
}
```

Sample Input:-

Enter elements of 4x4 matrix:

1 5 3 4

9 2 8 6

7 11 0 13

4 15 12 10

OUPUT:-

The largest number in the matrix is: 15

C Program: Find Transpose of a Matrix

```
#include <stdio.h>

int main() {
    int a[10][10], transpose[10][10];
    int rows, cols, i, j;

    // Input number of rows and columns
    printf("Enter rows and columns of matrix: ");
    scanf("%d %d", &rows, &cols);

    // Input elements of matrix
    printf("Enter elements of the matrix:\n");
    for(i = 0; i < rows; i++) {
        for(j = 0; j < cols; j++) {
            scanf("%d", &a[i][j]);
        }
    }
```

```
    // Calculate transpose
    for(i = 0; i < rows; i++) {
        for(j = 0; j < cols; j++) {
            transpose[j][i] = a[i][j];
        }
    }

    // Display transpose
    printf("Transpose of the matrix:\n");
    for(i = 0; i < cols; i++) {
        for(j = 0; j < rows; j++) {
            printf("%d\t", transpose[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

SAMPLE INPUT:-

Enter rows and columns of matrix: 2 3
Enter elements of the matrix:
1 2 3
4 5 6

OUTPUT:-

Transpose of the matrix:

1	4
2	5
3	6

Q: What is string? (2015-16)

Ans: In **C language**, a **string** is just a **collection of characters** stored together in a **character array** and **ends with a special character** called the **null character.**

Function	Description	Syntax
strlen()	It is used to find the length of the string.	char S[20] = "Hello";n = strlen(S);
strcpy()	It is used to copy string S2 to S1.	char S1[10], S2[10] = "Hello";strcpy(S1, S2);
strcat()	It is used to concatenate two strings and store result in S1.	char S1[10] = "Hello";char S2[10] = "Hai";strcat(S1, S2);
strcmp()	It is used to compare two strings.	char S1[10] = "Hai", S2[10];int i;i = strcmp(S1, S2);
strrev()	It is used to reverse the string.	char S[10] = "Hello";strrev(S);

Program to check string is palindrome or not

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main() {
   char s[10], s1[10];
   clrscr();
   printf("Enter String:\n");
   gets(s);
   strcpy(s1, s);
   strrev(s1);

   if (strcmp(s, s1) == 0)
      printf("I'm Palindrome\n");
   else
      printf("I'm Not Palindrome\n");
   getch();
}
```

Program to sort string in alphabetical order

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main() {
   char a[10], t;
   int i, j;
   clrscr();
   printf("Enter string:\n");
   gets(a);
   for (i = 0; i < strlen(a); i++) {
      for (j = i + 1; j < strlen(a); j++) {
         if (a[i] > a[j]) {
            t = a[i];
            a[i] = a[j];
            a[j] = t;
         }
      }
   }
   printf("In sorted string is: %s", a);
   getch();
}
```

Q: Explain structure with suitable example (2015-16). Difference between structure & array (2015-16, 2016-17)

Array

1. Array is the collection of same data type elements.
2. Elements are stored in contiguous memory location.
3. Elements are accessed by their index.
4. Array declaration and initialization:
   int A[3] = {10, 20, 30};
5. Every element is array of same size.
6. No keyword is used.

Structure

1. It is the collection of variables of different data types.
2. Elements may not be stored in a contiguous memory location.
3. Elements are accessed by their names.Elements are accessed using operators i.e., .
4. (Dot operator)Every element in a structure is of different data type.
5. struct keyword is used.

Syntax:

```
struct structname
{ data type element1;
data type element2; };
structurename variable;
```

WAP to check whether two dates are equal or not using structure

```
#include <stdio.h>
#include <conio.h>
void main() {
   struct date {
      int d, m, y;
   };
   struct date d1, d2;
   clrscr();
   printf("Enter First Date\n");
   scanf("%d-%d-%d", &d1.d, &d1.m, &d1.y);
   printf("Enter Second Date\n");
   scanf("%d-%d-%d", &d2.d, &d2.m, &d2.y);
   if (d1.d == d2.d && d1.m == d2.m && d1.y == d2.y)
      printf("\nEqual Dates\n");
   else
      printf("\nUnequal Dates\n");
   getch();
}
```

WAP to check whether Time1 is equal to Time2 or not using structure.

```
#include <stdio.h>
#include <conio.h>
void main() {
   struct time {
      int h, m, s;
   };
   struct time t1, t2;
   clrscr();
   printf("Enter Time one:\n");
   scanf("%d-%d-%d", &t1.h, &t1.m, &t1.s);
   printf("Enter Time two:\n");
   scanf("%d-%d-%d", &t2.h, &t2.m, &t2.s);
   if (t1.h == t2.h && t1.m == t2.m && t1.s == t2.s)
      printf("\nEqual Time\n");
   else
      printf("\nUnequal Time\n");
   getch();
}
```

Program to store record of 100 students & print record whose marks > 75.

```
#include <stdio.h>
#include <conio.h>
void main() {
   struct student {
      int sno;
      char name[10];
      int marks;
   };
   struct student s[100];
   int i;
   printf("Enter details of students:\n");
   for (i = 0; i < 100; i++) {
      scanf("%d %s %d", &s[i].sno, s[i].name, &s[i].marks);
   }
   printf("Students whose record marks > 75:\n");
   for (i = 0; i < 100; i++) {
      if (s[i].marks > 75) {
         printf("\n%d %s %d", s[i].sno, s[i].name, s[i].marks);
      }
   }
   getch();
}
```

Create a suitable structure in C for keeping the record of employees. Display those with salary > 20,000.

```
#include <stdio.h>
#include <conio.h>
struct emp {
    int eid;
    char en[20];
    char desig[10];
    char dept[10];
    char city[10];
    int esal;
};
void main() {
    struct emp e[100];
    int i;
    printf("Enter details of 100 employees:\n");
    for (i = 0; i < 100; i++) {
        scanf("%d %s %s %s %s %d", &e[i].eid, e[i].en, e[i].desig, e[i].dept, e[i].city, &e[i].esal);
    }
```

```
printf("Name of employees whose salary is above 20000:\n");
   for (i = 0; i < 100; i++) {
      if (e[i].esal > 20000) {
         printf("\n%s", e[i].en);
      }
   }
   getch();
}
```

Differentiate between Structure and Union

Structure

1. Keyword struct is used to define a structure.
2. Each member within a structure is assigned a unique memory location.
3. Changing the value of one data member will not affect other data members in the structure.
4. The total size depends on the sum of the sizes of all data members.
5. We can access any member at a time.

Syntax:

```
struct student {
    char name[20];
    int age;
    char address[50];
};
struct student s1;
```

Union

1. Keyword union is used to define a union.
2. Memory location is shared by all data members.
3. Changing the value of one data member will change the value of other data members.
4. The total size depends only on the size of the largest data member.
5. Only one member can be accessed at a time.

Syntax:

```
union student {
   char name[20];
   int age;
   char address[50];
};
union student s1;
```

What is Sorting?

Sorting means arranging data in a particular order either ascending (small to large) or descending (large to small).
It is commonly used in computer science to organize data so it's easier to search, analyze, or display.

Commonly Types or Methods Of Sorting:-

1. Bubble Sort
2. Selection Sort
3. Insertion Sort

Program of Bubble Sort

```
#include <stdio.h>
#include <conio.h>

void main()
{
   int a[25], n, j, i, t;

   printf("Enter no of elements \n");
   scanf("%d", &n);

   printf("Enter array elements \n");
   for(i = 0; i < n; i++)
        {
      scanf("%d", &a[i]);
         }
```

```
for(i = 0; i < n; i++)
  {
     for(j = 0; j < n - i - 1; j++)
     {
        if(a[j] > a[j + 1])
        {
           t = a[j];
           a[j] = a[j + 1];
           a[j + 1] = t;
        }
     }
  }

  printf("The sorted array is: \n");
  for(i = 0; i < n; i++)
        {
     printf("%d ", a[i]);
          }
  getch();
}
```

WAP to sort elements using Selection Sort (2018–19)

```
#include <stdio.h>
#include <conio.h>

void main()
{
   int a[25], n, j, i, t;
   clrscr();

   printf("Enter no of elements \n");
   scanf("%d", &n);

   printf("Enter each element \n");
   for(i = 0; i < n; i++)
       {
      scanf("%d", &a[i]);
           }
```

```
for(i = 0; i < n; i++)
  {
     for(j = i + 1; j < n; j++)
     {
        if(a[i] > a[j])
        {
           t = a[i];
           a[i] = a[j];
           a[j] = t;
        }
     }
  }

  printf("The sorted array is: \n");
  for(i = 0; i < n; i++)
     printf("%d ", a[i]);
  getch();
}
```

Program (Insertion Sort)

```
#include <stdio.h>
#include <conio.h>

void main()
{
   int i, j, a[25], n, key;

   printf("How many elements: ");
   scanf("%d", &n);

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

```
for(i = 1; i < n; i++)
 {
    key = a[i];
    for(j = i - 1; j >= 0; j--)
    {
       if(a[j] > key)
       {
          a[j + 1] = a[j];
       }
       else  {
          break;
       }
    }
    a[j + 1] = key;
 }
 printf("\nSorted array is:\n");
 for(i = 0; i < n; i++)
 {
    printf("%d ", a[i]);
 }
 getch();
}
```

Binary Search Technique in a Given Array

```
#include <stdio.h>
#include <conio.h>

void main()
{
   int i, a[30], n, e, l,f, mid, h;

   printf("Enter no. of elements\n");
   scanf("%d", &n);

   printf("Enter array elements\n");
   for(i = 0; i < n; i++)
     {
      scanf("%d", &a[i]);
       }
   printf("Enter number to search\n");
   scanf("%d", &e);
   f=0;
```

```
    l = n - 1;
    while (f <= l)
    {
        mid = (f + l) / 2;
        if(e == a[mid])
        {
            printf("%d is present at location %d\n", e, mid + 1);
            break;
        }
        else if(e > a[mid])
            f = mid + 1;
        else
            l = mid - 1;
    }

    if(f > l)
        printf("%d is not present\n", e);

    getch();
}
```

Linear Search Technique in a Given Array

```
#include <stdio.h>
#include <conio.h>

void main()
{
   int i, e, n, a[30];

   printf("Enter no. of elements\n");
   scanf("%d", &n);

   printf("Enter array elements\n");
   for(i = 0; i < n; i++)
      {
      scanf("%d", &a[i]);
         }
   printf("Enter number to search\n");
   scanf("%d", &e);
```

```
for(i = 0; i < n; i++)
   {
      if(a[i] == e)
      {
         printf("%d is present at location %d\n", e, i + 1);
         break;
      }
   }

   if(i == n)
      printf("%d is not present\n", e);

   getch();
}
```

Find Odd & Even Numbers from Array and Count

```
#include <stdio.h>
#include <conio.h>

void main()
{
   int i, n, even = 0, odd = 0, a[100];

   printf("Enter no. of elements\n");
   scanf("%d", &n);

   printf("Enter the elements\n");
   for(i = 0; i < n; i++)
    {
      scanf("%d", &a[i]);
      }
```

```
for(i = 0; i < n; i++)
   {
      if(a[i] % 2 == 0)
      {
         printf("%d is even\n", a[i]);
         even++;
      }
      else
      {
         printf("%d is odd\n", a[i]);
         odd++;
      }
   }

   printf("No. of odd numbers are: %d\n", odd);
   printf("No. of even numbers are: %d\n", even);

   getch();
}
```

THANK YOU FOR WATCHING

DO LIKE SHARE COMMENT AND SUBSCRIBE