MATRIX MULTIPLICATION:

```
#include <stdio.h>

int main() {     int
r1, c1, r2, c2;
   printf("Enter rows and columns of first matrix: ");
scanf("%d %d", &r1, &c1);     printf("Enter rows and
columns of second matrix: ");     scanf("%d %d", &r2,
&c2);

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

   int A[10][10], B[10][10], C[10][10];

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

   // Input second matrix     printf("Enter
elements of second matrix:\n");
   for (int i = 0; i < r2; i++) {
```

```
        for (int j = 0; j < c2; j++) {
scanf("%d", &B[i][j]);
        }
    }

    // Initialize result matrix with 0
for (int i = 0; i < r1; i++) {        for
(int j = 0; j < c2; j++) {
            C[i][j] = 0;
        }
    }

    // Matrix multiplication
    for (int i = 0; i < r1; i++) {
for (int j = 0; j < c2; j++) {
for (int k = 0; k < c1; k++) {
                C[i][j] += A[i][k] * B[k][j];
            }
        }
    }

    // Print result
printf("Resultant Matrix:\n");
for (int i = 0; i < r1; i++) {
for (int j = 0; j < c2; j++) {
printf("%d ", C[i][j]);
        }
        printf("\n");
    }

    return 0;
```

```
}
```

ODD OR EVEN NUMBERS:

```
#include <stdio.h>


int main() {     int num;
printf("Enter a number: ");
scanf("%d", &num);


   if (num % 2 == 0)
printf("%d is Even\n", num);
   else
      printf("%d is Odd\n", num);


   return 0;
}
```

FACTORIAL USING RECURSION:

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

```
// Recursive function to calculate factorial
int factorial(int n) {     if (n == 0 || n == 1)
// base case
       return 1;     else        return n * factorial(n
- 1);  // recursive call
}

int main() {     int num;
printf("Enter a number: ");
scanf("%d", &num);

   if (num < 0) {        printf("Factorial not defined for
negative numbers.\n");
   } else {
      printf("Factorial of %d = %d\n", num, factorial(num));
   }

   return 0;
}
```

FIBANOCCI SERIES USING RECURSION:

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

```
// Recursive function for Fibonacci int
fibonacci(int n) {     if (n == 0) return 0;   //
base case     if (n == 1) return 1;   // base
case     return fibonacci(n - 1) +
fibonacci(n - 2);
}

int main() {
   int n, i;
   printf("Enter the number of terms: ");
scanf("%d", &n);

   printf("Fibonacci Series: ");
   for (i = 0; i < n; i++) {
printf("%d ", fibonacci(i));
   }
   printf("\n");

   return 0;
}
```

FACTORIAL WITHOUT USING RECURSION:

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

```
int main() {     int num, i;     unsigned long long fact =
1;  // factorial can be large

   printf("Enter a number: ");
scanf("%d", &num);

   if (num < 0) {        printf("Factorial not defined for
negative numbers.\n");
   } else {        for (i = 1; i <= num; i++)
{           fact *= i;  // multiply i with
fact
       }
       printf("Factorial of %d = %llu\n", num, fact);
   }

   return 0;
}
```

FIBONACCI SERIES USING RECURSION:

```
#include <stdio.h>

// Recursive function to return nth Fibonacci number
int fibonacci(int n) {     if (n == 0) return 0;
// base case     if (n == 1) return 1;   //
```

```
base case     return fibonacci(n - 1) +
fibonacci(n - 2);
}

int main() {
   int n, i;
   printf("Enter number of terms: ");
scanf("%d", &n);

   printf("Fibonacci Series: ");
   for (i = 0; i < n; i++) {
printf("%d ", fibonacci(i));
   }
   printf("\n");

   return 0;
}
```