Exercise 1: Write a program to convert English units to metric (i.e., miles to kilometers, gallons to liters, etc.). Include a specification and a code design.

```
#include<stdio.h>
void mile_km(float);
void gallons_litre(float);

int main(){

   while (1)
   {
      int op;
      float value;
      printf("Enter the choice\n1.MILE - KM\n2.GALLONS - LITRE\n3.EXIT\n");
      scanf("%d",&op);

      switch (op)
      {
      case 1:
         printf("Enter miles: ");
         scanf("%f", &value);
         mile_km(5);
         break;
      case 2:
         printf("Enter gallons: ");
         scanf("%f", &value);
         gallons_litre(5);
         break;
      case 3:
         printf("Exiting program.\n");
```

```
            return 0;


        default:
        printf("invalid option\n");
            break;
        }
    }

}

void mile_km(float mile){
    printf("%2.f miles equals to %2.f kilometers\n",mile,mile*1.60934);


}
void gallons_litre(float gal){
    printf("%.2f gallons equals to %.2f litres\n",gal,gal*3.78541);


}
```

Exercise 2: Write a program to perform date arithmetic such as how many days there are between 6/6/90 and 4/3/92. Include a specification and a code design.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h> // Include for bool type
struct date {
int day;
int month;
int year;
};
bool isLeapYear(int year);
```

```
int daysInMonth(int month, int year);
int date_arithmetic(struct date d);
int main() {
struct date d1, d2;
printf("Enter the starting date (dd-mm-yyyy): \n");
scanf("%d-%d-%d", &d1.day, &d1.month, &d1.year);
printf("Enter the ending date (dd-mm-yyyy): \n");
scanf("%d-%d-%d", &d2.day, &d2.month, &d2.year);
int total_days1 = date_arithmetic(d1);
int total_days2 = date_arithmetic(d2);
int difference = abs(total_days1 - total_days2);
printf("The number of days between the two dates is %d.\n", difference);
return 0;
}

// Function to check if a year is a leap year
bool isLeapYear(int year) {
return (year % 400 == 0) || (year % 100 != 0 && year % 4 == 0);
}
// Function to calculate the number of days in a given month
int daysInMonth(int month, int year) {
switch (month) {
case 4: case 6: case 9: case 11:
return 30;
case 2:
return isLeapYear(year) ? 29 : 28;
default:
return 31;
}
}
// Function to convert a date to the number of days since 1/1/0000
```

```
int date_arithmetic(struct date d) {
int days = d.day;
for (int y = 0; y < d.year; y++) {
days += isLeapYear(y) ? 366 : 365;
}
for (int m = 1; m < d.month; m++) {
days += daysInMonth(m, d.year);
}
return days;
}
```

Exercise 3: A serial transmission line can transmit 960 characters each second. Write a program that will calculate the time required to send a file, given the file's size. Try the prog ram on a 400MB (419,430,400 -byte) file. Use appropriate units. (A 400MB file takes days.)

/*A serial transmission line can transmit 960 characters each second.

Write a program that will calculate the time required to send a file, given the file's

size. Try the program on a 400MB (419,430,400 -byte) file. Use appropriate units.

(A 400MB file takes days.)*/

```
#include <stdio.h>
int main() {
long long fileSizeInBytes = 419430400; // 400 MB in bytes
int transmissionRate = 960; // 960 characters per second (1 byte = 1 character)
// Calculate the total time in seconds
long long totalTimeInSeconds = fileSizeInBytes / transmissionRate;
// Convert time into days, hours, minutes, and seconds
int days = totalTimeInSeconds / (24 * 3600);
totalTimeInSeconds %= (24 * 3600);
int hours = totalTimeInSeconds / 3600;
totalTimeInSeconds %= 3600;
int minutes = totalTimeInSeconds / 60;
int seconds = totalTimeInSeconds % 60;
printf("The time required to send a 400MB file is: %d days, %d hours, %d minutes, and %d
seconds.\n", days, hours, minutes, seconds);

return 0;

}
```

Exercise 4: Write a program to add an 8% sales tax to a given amount and round

the result to the nearest penny.

```
#include<stdio.h>
#include<math.h>
int main(){
  float tax,amount,total;
  printf("Enter amount");
  scanf("%f",&amount);
  tax=amount*0.08;
  total = amount+tax;
  total=round(total*100)/100;
  printf("Amount : %.2f\n",amount);
  printf("Sales Tax (8%%): %.2f\n",tax);
  total=
  printf("Total Amount = %.2f\n",total);
}
```

Exercise 5: Write a program to tell if a number is prime.

```
#include<stdio.h>
int main(){
  int num,flag=1;
  printf("Enter a NUmber");
  scanf("%d",&num);
```

```
    if(num<=1){
        flag=0;
    }else{
          for(int i=2;i<=num/2;i++){
          if(num%i==0){
              flag=0;


          }
    }


    }


    if(flag){
            printf("%d is a prime number",num);
        }else{
            printf("%d not a prime number",num);
        }
}
```

Exercise 6: Write a program that takes a series of numbers and counts the number of positive and negative values.

```
#include<stdio.h>
int main(){


    int n;
    int n_count=0,p_count=0;
    printf("Enter number of elements");
     scanf("%d",&n);
    int arr[n];
    for(int i=0;i<n;i++){
        printf("Element %d = ",i+1);
        scanf("%d",&arr[i]);

    }
    for(int i=0;i<n;i++){
        if(arr[i]<0){
            n_count+=1;

        }else
        {
          p_count+=1;
        }

    }
    printf("Count of posiitive number : %d\n",p_count);
     printf("Count of negative number : %d\n",n_count);
}
```

C program to find HCF of given numbers using recursion

```
#include<stdio.h>
int HCF_recursion(int,int);
int main(){
    int num1,num2;
    printf("Enter the 2 number: ");
    scanf("%d%d",&num1,&num2);
    int res=HCF_recursion(num1,num2);
    printf("HCF = %d",res);
}
```

```
int HCF_recursion(int a,int b){
    if(b==0){
        return a;
    }return HCF_recursion(b,a%b);

}
```

C program to find LCM of give numbers using recursion

```
#include<stdio.h>
int LCM_recursion(int,int);
int HCF_recursion(int,int);
int main(){
    int num1,num2;
    printf("enter 2 numbres: ");
    scanf("%d%d",&num1,&num2);
    int res=LCM_recursion(num1,num2);
    printf("LCM = %d",res);
}

int HCF_recursion(int a,int b){
    if(b==0){
        return a;
    }
   return HCF_recursion(b,a%b);

}
int LCM_recursion(int a,int b){
    int hcf=HCF_recursion(a,b);
    return(a*b)/hcf;
}
```

C program to convert a decimal to binary using recursion

```
#include<stdio.h>
void decimal_ToBinary(int);
int main(){
   int num;
   printf("enter number");
   scanf("%d",&num);
   printf("Equivalant binary :");
   decimal_ToBinary(num);



}
void decimal_ToBinary(int n){
   if(n>1){
      decimal_ToBinary(n/2);
   }printf("%d",n%2);
}
```

C program to convert Binary to gray code

```
#include<stdio.h>
void Binary_Gray(int);
int main(){
   int num;
   printf("enter number");
   scanf("%d",&num);
   printf("Gray code = ");
   Binary_Gray(num);
}

void Binary_Gray(int n){
   int gray=n^(n>>1);
   printf("%d",gray);
}
```

print following pyramid

```
********
***  ***
**    **
*      *
```

```
#include <stdio.h>

int main() {
    int i, j;
    for (i = 0; i < 4; i++) {
        for (j = 0; j < 8 - i; j++) {
            printf("*");
        }
        for (j = 0; j < 2 * i - 1; j++) {
            printf(" ");
        }
        if (i > 0) {
            for (j = 0; j < 8 - i; j++) {
                printf("*");
            }
        }
        printf("\n");
    }

    return 0;
}
```

C program to find the sum of Natural Number/Factorial of Number

```
#include <stdio.h>

float factorial(int n) {
    float fact = 1;
    for (int i = 1; i <= n; i++) {
        fact *= i;
    }
    return fact;
}

float series_sum(int n) {
    float sum = 0;
    for (int i = 1; i <= n; i++) {
        sum += (float) i / factorial(i);
    }
    return sum;
}

int main() {
    int n;
    printf("Enter a positive integer: ");
    scanf("%d", &n);
    printf("Sum of the series: %.2f\n", series_sum(n));
    return 0;
}
```

C program to find sum of the given series

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

float series_sum(int n) {
    float sum = 0;
    for (int i = 1; i <= n; i++) {
        int term = 2 * i - 1;
        sum += pow(term, 2) / pow(term, 3);
    }
    return sum;
}

int main() {
    int n;
    printf("Enter the number of terms: ");
    scanf("%d", &n);
    printf("Sum of the series: %.2f\n", series_sum(n));
    return 0;
}
```

Replace EVEN elements with 0 and ODD with 1 in One Dimensional Array

```
#include <stdio.h>

int main() {
    int n;

    printf("Enter the size of the array: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter the elements of the array:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
        arr[i] = (arr[i] % 2 == 0) ? 0 : 1;
    }

    printf("Modified array:\n");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    return 0;
}
```

c program to read a matrix and print diagonals

```
#include <stdio.h>
int main() {
int n;
printf("Enter the size of the matrix (n x n): ");
scanf("%d", &n);
int matrix[n][n];
printf("Enter the elements of the %d x %d matrix:\n", n, n);
for (int i = 0; i < n; i++) {
for (int j = 0; j < n; j++) {
scanf("%d", &matrix[i][j]);
}
}
printf("The diagonals are: ");
for (int i = 0; i < n; i++) {
for (int j = 0; j < n; j++) {
if(i == j){
printf("%d ",matrix[i][j]);
}
}
}
}
```

C program to print the upper triangular portion of a 3*3 matrix

```
#include <stdio.h>
int main() {
int matrix[3][3];
printf("Enter the elements of the 3x3 matrix:\n");
for (int i = 0; i < 3; i++) {
for (int j = 0; j < 3; j++) {
scanf("%d", &matrix[i][j]);
}
}
printf("Upper triangular portion of the matrix:\n");
for (int i = 0; i < 3; i++) {
for (int j = 0; j < 3; j++) {
if (i <= j) {
printf("%d ", matrix[i][j]);
} else {
printf(" ");
}
}
printf("\n");
} return 0; }
```

. **Input and Print Text using Dynamic Memory Allocation**

```
#include <stdio.h>

#include <stdlib.h>


int main() {

    char *text;

    int size;


    printf("Enter the size of the text: ");

    scanf("%d", &size);


    text = (char *)malloc((size + 1) * sizeof(char)); // +1 for null terminator

    if (text == NULL) {

        printf("Memory allocation failed!\n");

        return 1;

    }


    printf("Enter the text: ");

    scanf(" ");

    fgets(text, size + 1, stdin);


    printf("You entered: %s\n", text);


    free(text);

    return 0;

}
```

Sum of Elements in Array with Dynamic Memory Allocation

```
#include <stdio.h>

#include <stdlib.h>
```

```
int main() {
    int n, *arr, sum = 0;

    printf("Enter the size of the array: ");
    scanf("%d", &n);

    arr = (int *)malloc(n * sizeof(int));
    if (arr == NULL) {
        printf("Memory allocation failed!\n");
        return 1;
    }

    printf("Enter the elements of the array:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
        sum += arr[i];
    }

    printf("Array elements: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }

    printf("\nSum of elements: %d\n", sum);

    free(arr);
    return 0;
}
```