```c
#include <stdio.h>
#include <stdlib.h>
#define MAX 100

struct node
{
    int item;
    struct node *link;
};

void baseconv(int num, int base)
{
    int rem = num % base;
    if (num == 0)
        return;
    baseconv(num / base, base);

    if (rem < 10)
        printf("%d", rem);
    else
        printf("%c", (rem - 10) + 'A');
}

void hanoi(int n, char source, char temp, char dest)
{
    if (n == 1)
    {
        // only one disk present
        printf("Move %d disk from %c --> %c\n", n, source, dest);
        return;
    }
    hanoi(n - 1, source, dest, temp);
    printf("Move %d disk from %c --> %c\n", n, source, dest);
    hanoi(n - 1, temp, source, dest);
}

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

void reverse(char expression[])
{
    if (*expression == '\0')
        return;
    reverse(expression + 1);
    putchar(*expression);
}

struct node *addatbeg(struct node *start, int data)
{
    struct node *tmp;
    if ((tmp = (struct node *)malloc(sizeof(struct node))) == NULL)
    {
        printf("No memory available\n");
        exit(EXIT_FAILURE);
    }
```

```
    tmp->item = data;
    tmp->link = start;
    start = tmp;
    return start;
}

struct node *addatend(struct node *start, int data)
{
    struct node *p = start, *tmp;
    while (p->link != NULL)
        p = p->link;
    if ((tmp = (struct node *)malloc(sizeof(struct node))) == NULL)
    {
        printf("Not enough Memory\n");
        exit(EXIT_FAILURE);
    }
    tmp->link = NULL;
    tmp->item = data;
    p->link = tmp;
    return start;
}

struct node *create(struct node *start)
{
    int n, data;
    printf("Enter the number of nodes in the Linked List: ");
    scanf("%d", &n);
    printf("Enter the data: ");
    scanf("%d", &data);
    start = addatbeg(start, data);
    for (int i = 2; i <= n; i++)
    {
        int tdata;
        printf("Enter the data: ");
        scanf("%d", &tdata);
        start = addatend(start, tdata);
    }
    printf("Linked List w %d elements was successfulyl created!\n", n);
    return start;
}

int search(struct node *start, int item)
{
    struct node *tmp = start;
    if (tmp == NULL)
        return 0;
    if (tmp->item == item)
        return 1;
    search(tmp->link, item);
}

int main(int argc, char const *argv[])
{
    while (1)
    {
    here:
        printf("\nEnter choice\n1. Base Conversion\n2. Tower of Hanoi\n3. Greatest
Common Divisor\n4. Reverse a String\n5. Search an Item in a Linked List\n");
        int choice;
```

```
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
        {
            printf("Enter the number in decimal: ");
            int num;
            scanf("%d", &num);
            while (1)
            {
                printf("\nChoose\n1. Dec to Bin\n2. Dec to Oct\n3. Dec to Hex\n4.
Dec to Duodecimal");
                printf("\n");
                int i;
                scanf("%d", &i);
                switch (i)
                {
                case 1:
                    printf("\nThe Binary equivalent of %d is: ", num);
                    baseconv(num, 2);
                    printf("\n");
                    break;
                case 2:
                    printf("\nThe Octal equivalent of %d is: ", num);
                    baseconv(num, 8);
                    printf("\n");
                    break;
                case 3:
                    printf("\nThe Hexadecimal equivalent of %d is: ", num);
                    baseconv(num, 16);
                    printf("\n");
                    break;
                case 4:
                    printf("The Duodecimal equivalent of %d is: ", num);
                    baseconv(num, 12);
                    printf("\n");
                    break;
                default:
                    goto here;
                }
            }
            break;
        }

        case 2:
        {
            char source = 'A', temp = 'B', dest = 'C';
            printf("Enter the number of disks: ");
            int n;
            scanf("%d", &n);
            hanoi(n, source, temp, dest);
            break;
        }

        case 3:
        {
            int a, b;
            printf("Enter first nuber: ");
            scanf("%d", &a);
```

```
            printf("Enter second number: ");
            scanf("%d", &b);
            printf("GCD is: %d\n", gcd(a, b));
            break;
        }

        case 4:
        {
            char expression[MAX];
            getchar();
            printf("Enter the string to be reversed: ");
            gets(expression);
            reverse(expression);
            printf("\n");
            break;
        }

        case 5:
        {
            struct node *start;
            start = create(start);
            int data;
            printf("Enter the item to be searched: ");
            scanf("%d", &data);
            if (search(start, data))
                printf("%d is present in the Linked List\n", data);
            else
                printf("%d is not present in the Linked List\n", data);
            break;
        }
        default:
            exit(EXIT_SUCCESS);
            break;
        }
    }
    return 0;
}
```