

```

include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
void Exit(const char *err);
typedef struct Literal_t {
    void *Next;
    int m;
    int a,b;
    int X; } Literal_t;
typedef struct Term_t {
    int cnst;
    Literal_t *First,*Last; } Term_t;
void AddLiteral(Term_t *t,int m,int a,int b)
{
    Literal_t *l;
    l = (Literal_t*)malloc(sizeof(Literal_t));
    if (!l) Exit("No room for literal");
    if (t->First) t->Last->Next = l;
    else t->First = t->Last = l;
    l->Next = 0;
    l->m = m;
    l->a = a;
    l->b = b;
    t->Last = l;
}
void DelTerm(Term_t *t)
{
    Literal_t *l = t->First,*l1;
    while(l) { l1 = (Literal_t*) l->Next;
                free(l);
                l = l1;
            }
    t->First = t->Last = 0;
}
void PrintTerm(Term_t *t)
{
    Literal_t *l = t->First;
    int I = 0;
    printf("%d ",t->cnst);
    while (l) {
        printf("* X%d(%d,%d)",++I,l->a,l->b);
        l = (Literal_t*) l->Next;
    }
    puts("");
}
int LiteralValue(Literal_t *l)
{
    return (l->X<l->a || l->X>l->b) ? 0:l->m-1;
}
int TermValue(Term_t *t)

```

```

{
    Literal_t *l = t->First;
    int value = t->cnst;
    while (l) {
        value = (LiteralValue(l)<value)? LiteralValue(l):value;
        l = (Literal_t*) l->Next;
    }
    return value;
}
int TermValues(Term_t *t)
{
    Literal_t *l = t->First;
    int value = t->cnst;
    while (l) {
        value = (l->m-1<value)? l->m-1:value;
        l = (Literal_t*) l->Next;
    }
    return value;
}
void PrintTermValues(Term_t *t)
{
    Literal_t *l = t->First;
    int I = 0,j;
    printf("Values: %d for",TermValues(t));
    while (l) {
        printf("  X%d = ",++i);
        for(j = l->a; j<l->b; j++)
            printf("%d,",j);
        printf("%d",j);
        l = (Literal_t*) l->Next;
        if (l) printf(", and ");
    }
    printf("\n      0 otherwise");
}
int main()
{
    Term_t *term [10],*t;
    int n_terms,n_literals;
    int i,j;
    int m,a,b;
    Literal_t *tl;
    clrscr();
    printf("Input number of terms: ");
    scanf("%d",&n_terms);
    if (!n_terms || n_terms>10) Exit("Number of terms from 1 to 10");
    for (i = 0; i<n_terms; i++)
    {
        t = (Term_t*)malloc(sizeof(Term_t));
        if (!t) Exit("No room for term");
        t->First = t->Last = 0;
        printf("Term%d: \n",i+1);
        printf("Input constant: ");
    }
}

```

```

scanf("%d",&t->cnst);
printf("Input number of literals: ");
scanf("%d",&n_literals);
term[i] = t;
for(j = 0; j<n_literals; j++)
{
    printf("Input m,a,b for the literal %d: ",j+1);
    scanf("%d,%d,%d",&m,&a,&b);
    AddLiteral(t,m,a,b);
}
return 0;
}
for(;;)
{
    for(i = 0; i<n_terms; i++)
    {
        printf("\n Term%d: ",i+1);
        PrintTerm(term[i]);
        for(tl = term[i]->First,j = 0; tl; tl = tl->Next)
        {
            printf("Input x%d: ",j+1);
            scanf("%d",&tl->X);
            j++;
        }
        printf("Value: %d",TermValue(term[i]));
    }
    printf("\n One more term (y/n)?");
    j = getch();
    if (j=='n' || j=='N') break;
}
for (I = 0; i<n_terms; i++)
{
    DelTerm(term[i]);
    free(term[i]);
    term[i] = 0;
}
puts("\nPress any key...");
getch();
}
void Exit(const char *err)
{
    printf("Error: %s\n",err);
    exit(1);
}

```