```
#include <stdio.h>

#define MAXDIM 50
#define TRUE 1
#define FALSE 0


   typedef enum { Perdedora, Tablas, Ganadora } TPosicion;

   typedef struct {
       int Peso, Valor, Cogido;
    } TObjeto;

    typedef struct {
        int Peso, Valor;
    } TJugador;

   void Evaluar (TJugador JugadorA, TJugador JugadorB, int K, TObjeto Objeto [], int
N,
                 TPosicion *Pos, int *NumObjeto);

   int PuedeCoger (TJugador Jugador, int K, TObjeto Objeto[], int N);

   void Entrada (int *K, int *N, TObjeto Objeto[]);

   void Inicializa(TJugador *JugadorA, TJugador *JugadorB, TObjeto Objeto[], int N);


main () {

   int N, K, i;
   TPosicion Pos;
   int NumObjeto;
   TJugador JugadorA, JugadorB;
   TObjeto Objeto[MAXDIM];


   Entrada (&K,&N,Objeto);
   Inicializa(&JugadorA,&JugadorB,Objeto,N);

   printf("\n\nEvaluando Posicion...\n");

   Evaluar (JugadorA,JugadorB,K,Objeto,N,&Pos,&NumObjeto);

   printf("\n\nLos objetos con su peso y valor son: \n");
   for (i = 0; i < N; i++)
      printf("Objeto %d -> Valor: %2d (Peso %2d)\n",i,Objeto[i].Valor,Objeto[i].Peso)
;

   printf("\n\nLa posicion del juego es ");

   if (Pos == Perdedora)
    printf("PERDEDORA\n\n");
```

```
   else {
    if (Pos == Ganadora) printf("GANADORA\n\n");
    else printf("TABLAS\n\n");

      printf("El resultado se obtiene cogiendo el objeto %d: (Valor: %d, Peso: %d)\n"
,
            NumObjeto,Objeto[NumObjeto].Valor,Objeto[NumObjeto].Peso);
      printf("(Los objetos comienzan a numerarse por 0)\n");
   }
} // fin main



   void Entrada (int *K, int *N, TObjeto Objeto[]) {
    int i;

      printf("Peso Maximo... "); scanf(" %d",K);
      printf("Numero de Objetos (< %d) ... ",MAXDIM); scanf(" %d",N);
      printf("Peso de cada objeto -> ");
      for (i=0; i<*N; i++) scanf(" %d", &Objeto[i].Peso);
      printf("\nValor de cada objeto -> ");
      for (i=0; i<*N; i++) scanf(" %d", &Objeto[i].Valor);
   }


   void Inicializa(TJugador *JugadorA, TJugador *JugadorB, TObjeto Objeto[], int N) {
    int i;
    JugadorA->Peso = 0; JugadorA->Valor = 0;
    JugadorB->Peso = 0; JugadorB->Valor = 0;
    for (i=0; i<N; i++) Objeto[i].Cogido = FALSE;
   }


   void Evaluar (TJugador JugadorA, TJugador JugadorB, int K, TObjeto Objeto[], int N
,
                 TPosicion *Pos, int *NumObjeto) {
      int i, x, Valor;
      TPosicion PosDejamos;

      if (!PuedeCoger(JugadorA,K,Objeto,N))
         if (!PuedeCoger(JugadorB,K,Objeto,N))
            // Ninguno de los dos coge pues se pasa del lÌmite
            if (JugadorA.Valor > JugadorB.Valor) *Pos = Ganadora;
            else if (JugadorA.Valor == JugadorB.Valor) *Pos = Tablas;
            else *Pos = Perdedora;
         else {
            // A pasa el turno pues se pasarÌa, y B puede seguir cogiendo
            Evaluar (JugadorB,JugadorA,K,Objeto,N,&PosDejamos,&x);
            if (PosDejamos == Ganadora) *Pos = Perdedora;
            else if (PosDejamos == Perdedora) *Pos = Ganadora;
            else *Pos = Tablas;
            *NumObjeto = -1; // Para indicar que no hacemos nada
         }
```

```
   else {
     // A puede coger, asÌ coger·
      i = 0; *Pos = Perdedora;
      while ((i<N) && (*Pos != Ganadora)) {
         if (!Objeto[i].Cogido && (JugadorA.Peso + Objeto[i].Peso <= K)) {
            Objeto[i].Cogido = TRUE;
            JugadorA.Valor += Objeto[i].Valor;
            JugadorA.Peso += Objeto[i].Peso;

            Evaluar (JugadorB,JugadorA,K,Objeto,N,&PosDejamos,&x);
            if (PosDejamos != Ganadora) {
             *NumObjeto = i;
                 if (PosDejamos == Perdedora) *Pos = Ganadora;
                 else *Pos = Tablas;
            }
            // Restablecemos los valores
            Objeto[i].Cogido = TRUE;
            JugadorA.Valor -= Objeto[i].Valor;
            JugadorA.Peso -= Objeto[i].Peso;
         }
         i++;
      }
   }
} // Fin Evaluar


int PuedeCoger (TJugador Jugador, int K, TObjeto Objeto[], int N) {
 int i;
 for (i= 0; i<N; i++)
    if (!Objeto[i].Cogido && (Jugador.Peso + Objeto[i].Peso <= K)) return TRUE;
 return FALSE;
}
```