```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define CAN_MAX 400
#define Componentes "Componentes.bin"
#define Paquetes "Paquetes.bin"
#define ProductosFinales "ProductosFinales.bin"
#define ARCHIVOSALIDA "Perdidas.bin"


typedef struct {
    int cantMateria; //gramos
    char calidad; // "a", "b", "c"
    float costo; //cada bloque posee distintos costos en base a los gramos
}PaqueteMateriaPrima;

typedef struct{
    char nombre[20]; //"Biblia", "Copa de vino", "Doll"
    int falla; //puede tener 0...2 fallas
    float costo;
    char calidad; //según la calidad de la materia, será la calidad del producto
   }Componente;


typedef struct{
    float costo;
    char calidad;
}PlayMobil;


///------------------------------PROTIPADO-----------------------------------------------
------------------------------
//void despersistencia(Componente comp[], int* validos);
void mostrarUnComponente(Componente componente);
//void mostrarComponentes(Componente componente[],int validos);
void mostrarComponentes(Componente doll[], Componente copaDeVino[], Componente biblia[],
int validos, int validos2, int validos3);
void despersistencia(Componente componentes[],int * validos);
void mostrar (Componente comp [], int validos);
void comparar(Componente componentes[], Componente doll[],  Componente copaDeVino[],
Componente biblia[],int validos, int *j, int *k, int* l);
void DespersistirPerdidas();
float costoEnsamble(Componente comp);
void crearProducto(Componente doll[], Componente biblia[], Componente copaDeVino[],
PlayMobil productoFinal[], int i, int j, int k, int l);
void persistencia(PlayMobil play[], int validos);
int crearPlaymobils (Componente arreglo1[], int validos1, Componente arreglo2[], int
validos2, Componente arreglo3[], int validos3, PlayMobil arregloPlaymobil[]);
void mostrarPlayMobil(PlayMobil play[], int validosPlayMobil);



int main()
{

    Componente componente[CAN_MAX];
    Componente doll[CAN_MAX];
    Componente biblia[CAN_MAX];
    Componente copaDeVino[CAN_MAX];


    PlayMobil productoFinal[CAN_MAX];

    int validos=0;
    int validos1=0;
```

```
    int validos2=0;
    int validos3=0;
    int valFloat=0;
    int validosPlayMobil=0;



    ///DESPERSISTIMOS EL ARCHIVO DE COMPONENTES EN UN ARREGLO GENERAL
    despersistencia(componente ,&validos);
    printf("MOSTRANDO TODOS LOS COMPONENTES \n");
    mostrar(componente, validos);
    printf("------------------------------------------------------------------ \n");

    ///COMPARAMOS EL ARREGLO GENERAL Y LO DIVIDIMOS EN TRES ARREGLOS, DESCARTANDO
AQUELLOS QUE TENGAN FALLAS >=2
    comparar(componente, doll, copaDeVino, biblia,validos, &validos1, &validos2, &
validos3);


    ///MOSTRAMOS LOS ARREGLOS
    printf("-----------------------------Doll------------------------------- \n");
    mostrar(doll, validos1);
    printf("-----------------------------Biblia------------------------------ \n");
    mostrar(biblia, validos3);
    printf("-----------------------------Copa DE vino------------------------- \n");
    mostrar(copaDeVino, validos2);

    ///DESPERSISTENCIA DE LAS PERDIDAS TOTALES
    DespersistirPerdidas();

    ///CREAMOS LOS PLAY MOBIL, Y MOSTRAMOS LA CANTIDAD QUE SE PUDIERON CREAR
    validosPlayMobil=crearPlaymobils(doll, validos1, copaDeVino, validos2, biblia,
validos3, productoFinal);
    printf("Cantidad de PlayMobil a Hacer: %d \n", validosPlayMobil);

    system("pause");

    ///MOSTRAMOS CADA PLAYMOBIL CREADO
    mostrarPlayMobil(productoFinal, validosPlayMobil);

    ///GUARDAMOS EL PRODUCTO FINAL EN UN ARCHIVO
    persistencia(productoFinal, validosPlayMobil);


    return 0;
}

void mostrarPlayMobil(PlayMobil play[], int validosPlayMobil)
{

    for(int i=0; i<validosPlayMobil; i++){
        printf("-------------PLAY MOBIL Num: %d ---------------------\n", i+1);
        printf("Costo: %f \n" ,play[i].costo);
        printf("Calidad: %c \n", play[i].calidad);
        printf("-----------------------------------------------\n");
    }

}

void despersistencia(Componente componentes[],int * validos)
{
    FILE* archi = fopen(Componentes,"rb");
    Componente p;


    if (archi)
```

```
    {
        while ((fread(&p,sizeof(Componente),1,archi) > 0) && *validos<CAN_MAX)
        {
            componentes[*validos] = p;
            (*validos)++;


        }


        fclose(archi);
    }
}

void comparar(Componente componentes[], Componente doll[],  Componente copaDeVino[],
Componente biblia[],int validos, int *j, int *k, int* l)
{
    int i=0;
    *j=0;
    *k=0;
    *l=0;
    FILE * archi= fopen(ARCHIVOSALIDA, "wb");


    if(archi)
    {
        while(i<validos)
        {

            if(componentes[i].falla >=2 )
            {
                fwrite(&componentes[i].costo, sizeof(float), 1, archi);


            }
            else
            {
                if(strcmp(componentes[i].nombre, "Doll")==0)
                {
                    doll[*j] =componentes[i];
                    (*j)++;

                }
                else if(strcmp(componentes[i].nombre, "Biblia")==0)

                {
                      biblia[*l]= componentes[i];
                        (*l)++;
                }
                else if(strcmp(componentes[i].nombre, "Copa de vino") ==0)
                {

                     copaDeVino[*k]=componentes[i];
                    (*k)++;

                }
            }

            i++;
        }
        fclose(archi);
    }
```

```
}



void mostrarUnComponente(Componente componente)
{
    printf("--------------------------------\n");
    printf("Nombre : %s\n",componente.nombre);
   // printf("Falla: %d\n", componente.falla);
    printf("Costo: $%f\n", componente.costo);
    printf("Calidad: %c\n", componente.calidad);
    printf("--------------------------------\n");
}


void mostrar (Componente comp [], int validos)
{
    for(int i=0; i<validos; i++)
    {

        mostrarUnComponente(comp[i]);
    }

}



void persistencia(PlayMobil play[], int validos)
{
    FILE * archi=fopen(ProductosFinales, "wb");

    int i=0;

    if(archi)
    {
        fwrite(&play[i], sizeof(PlayMobil), validos,archi);


        fclose(archi);
    }

}

void DespersistirPerdidas()
{
    printf("Despersistir perdidas\n");

    FILE* fp = fopen("perdidas.bin","rb");
    float perdidas = 0;
    float perdidastotales= 0;
    int cont=0;

    if(fp!=NULL)
    {
        printf("\nSe pudo leer el archivo de perdidas\n\n");

        while(fread(&perdidas,sizeof(float),1,fp)>0)
        {
            perdidastotales=perdidas+perdidastotales;
            cont++;
        }
    }
    else
    {
        printf("Hubo algun problema al cargar el archivo de perdidas, el proceso va a
terminar\n");
```

```
        system("pause");

    }

    fclose(fp);
    system ("pause");

    printf("Perdidas totales: $%f\n", perdidastotales);
    printf("Cantidad de perdidas totales: %d\n", cont);
}



int crearPlaymobils (Componente arreglo1[], int validos1, Componente arreglo2[], int
validos2, Componente arreglo3[], int validos3, PlayMobil arregloPlaymobil[])
{
    int cant=0,i=0,j,k,l=0;
    Componente aux;

    while (i<validos1)
    {
        j=0;
        k=0;

        while (arreglo2[j].calidad!=arreglo1[i].calidad&&j<validos2)
        {

            j++;

        }
        while (arreglo3[k].calidad!=arreglo1[i].calidad&&k<validos3)
        {

            k++;

        }
        if (j==validos2)
        {

            j=(-1);

        }
        if (k==validos3)
        {

            k=(-1);

        }
        if (j>=0 && k>=0)
        {

            arregloPlaymobil[l].calidad=arreglo1[i].calidad;
            switch(arreglo1[i].calidad)
            {


            case 'a':
                arregloPlaymobil[l].costo= 14;
                break;

            case 'b':
                arregloPlaymobil[l].costo= 12;
                break;

            case 'c':
```

```
                arregloPlaymobil[l].costo= 10;
                break;
            }
            arregloPlaymobil[l].costo=arregloPlaymobil[l].costo+arreglo1[i].costo+
arreglo2[j].costo+arreglo3[k].costo;
            l++;
            arreglo1[i].calidad='d';
            arreglo2[j].calidad='d';
            arreglo3[k].calidad='d';
            cant++;

//            aux=arreglo2[j];
//            arreglo2[j]=arreglo2[validos2-1];
//            arreglo2[validos2-1]=aux;
//            validos2--;
//            cant++;
//             aux=arreglo3[k];
//            arreglo3[k]=arreglo3[validos3-1];
//            arreglo3[validos3-1]=aux;
//            validos3--;
//            cant++;

        }
        i++;

    }


    return cant;
}
```