```cpp
#include <iostream>
#include <string>
#include <stdio.h>
#include "Timer.h"

using namespace std;


void muestra(int vec[]){

    for (int i=0; i<5 ;i++){
        cout<<vec[i]<<"\t";

    }
    cout<<endl;
}

void muestra(int vec[], int ini, int fin){
    for(;ini<=fin;ini++){
        cout<<vec[ini]<<"\t";
    }
    cout<<endl;
}

void Cambiar ( int i, int j, int *v ) {
    int x;
    x = v[i];
    v[i] = v[j];
    v[j] = x;
}

/**
 * @brief Función que reordena un vector respecto al parámetro 'piv'.
*/
int ordena(int vec[], int piv, int ini, int fin){
    int inf=ini;
    int sup=fin;
    int pivote=piv;     //Guardamos el valor del pivote
    int pospiv=-1;   //por si esta ordenado
    int aux=0;

    do{
        Cambiar(sup,inf,vec);
        while(vec[inf] <= pivote && inf<fin){
            if(vec[inf]==pivote)        //Si vemos que tornillo y tuerca encajan
                pospiv=inf;         //Guardamos su posición, para luego colocarla en su sitio.
            inf++;
        }

        while(vec[sup] >= pivote && sup>=ini){
            if(vec[sup]==pivote)        //Si vemos que tornillo y tuerca encajan
                pospiv=sup;         //Guardamos su posición, para luego colocarla en su sitio.
            sup--;
        }
    }while(inf < sup);


    if(pospiv-sup < 0)    //<Calculamos la posición en la que se quedará
        aux=sup;              //el pivote para que esté ordenado, ya que sup y inf puede.
    else
        if(pospiv-inf > 0) //que se encuentren en un mismo punto o queden cruzados
            aux=inf;
         else
            aux=pospiv;
    cout << sup <<","<<inf<<","<<pospiv <<","<<aux << endl;
   if (pospiv!=-1){  //Siempre se da esta condición ya que damos por hecho que hay los mismos tor. que
```

```
tuer.
        vec[pospiv]=vec[aux];       //<Colocamos el pivote en su sitio
        vec[aux]=pivote;
    }
    muestra(vec, ini, fin);   //mostramos tramo del vector en cuestion
    return aux;
}

/**
 * @brief Algoritmo de ordenación QuickShort aplicado al problema de tornillos y tuercas...
 */
void quicksort_Mod(int tor[],int tuer[], int ini, int fin){
    if(ini<=fin){
        int pivote1, pivote2;

        /**Primero ordenamos los tornillos con la primera tuerca
         * y guardamos en pivote1 la posición del tornillo que encaja con dicha tuerca... */
        cout<<endl<<"Cojemos la primera tuerca ("<<tuer[ini]<<") y ordenamos los tornillos: "<<endl;
        pivote1=ordena(tor, tuer[ini], ini, fin);

        cout<<"Posicion: "<<pivote1<<", para el tornillo: "<<tor[pivote1]<<endl;

        /**Ahora ordenamos las tuercas con el tornillo que encontramos antes
         * y guardamos en pivote2 la posición en la que queda la tuerca que hemos escogido antes...*/
        cout<<endl<<"Cojemos el anterior tornillo ("<<tor[pivote1]<<") y ordenamos las tuercas: "<<endl;
        pivote2=ordena(tuer, tor[pivote1], ini, fin);

        cout<<"Posicion Tuerca: "<<pivote2<<", con tornillo: "<<tor[pivote2]<<endl;

        /**Como todos los tornillos encajan con todas las tuercas, los dos que cojimos antes deben
         * haber quedado en la misma posición. Lo comprobamos:*/
        if(pivote1==pivote2) cout<<endl<<"Pareja encajada"<<endl;

        /** Y continuamos con la recurrencia... */
        quicksort_Mod(tor, tuer, ini, pivote1-1);
        quicksort_Mod(tor, tuer, pivote1+1, fin);
    }
}



int main(){
    int tornillos[]={3,6,4,1,2};
    int tuercas[]  ={4,1,2,3,6};

    Timer t;

 t.start();
    quicksort_Mod(tornillos, tuercas, 0, 4);
 t.stop();


    cout<<"Tuercas: ";
    muestra(tuercas);
    cout<<"\nTornillos: ";
    muestra(tornillos);
    cout << "El tiempo de ejecucion es de " ;
    cout  << t.getElapsedTimeInMilliSec() << " ms" << endl;

    return 0;


```