UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE CIENCIAS QUÍMICAS E INGENIERÍA

Ingeniería en Computación

Algoritmos y Estructura de Datos

Práctica 10:

Método de ordenamiento Quicksort

Catedrático: M.I. Alma Leticia Palacios Guerrero

Alumno: Reyes Udasco Rachelle Nerie

Matrícula: 1244220

Fecha de asignación: 30 de noviembre del 2018

Índice

Práctica No. 10

Método de ordenamiento Quicksort

1. Competencia

Determinar la eficiencia de un algoritmo de ordenación según su desempeño en escenarios de prueba con distintos parámetros, para ser considerada como criterio en la selección de un algoritmo para resolver un problema.

2. Marco Teórico

EL método quicksort es quizá el más utilizado, fue inventado en los 60's por C.A.R. Hoare, es un método popular porque no es difícil de implementar, funciona bien con diferentes tipos de datos y consume menos recursos que otros métodos.

Quicksort es un método divide y vencerás que funciona dividiendo un arreglo en dos partes, entonces ordena las partes de forma independiente, el punto de partición depende del orden inicial de los elementos en arreglo o archivo de entrada. El punto central de este método es el proceso de partición, en el cual se reacomoda el arreglo de acuerdo a las siguientes condiciones:

Ninguno de los elementos en a[l], ... a[i-1] es mayor que a[i].
Ninguno de los elementos en a[i+1], ... a[i-1] es menor que a[r].
Donde i es el pivote, l es izquierda y r derecha

El pseudocódigo recursivo es:

```
quickSort(arreglo[], low, high)
{
    if (low < high)
    {
        pivote = partition(arreglo, low, high);
        quickSort(arreglo, low, pi - 1);
        quickSort(arreglo, pi + 1, high);
    }
}
```

Y el pseudocódigo de la partición del arreglo es

```
partition (arreglo[], low, high)
{
    pivote = arreglo[high];
     i= (low - 1)
    for (j = low; j <= high- 1; j++)
    {
        if (arreglo[j] <= pivot)
        {
            i++;
            swap arreglo[i] and arreglo[j]
        }
    }
    swap arreglo[i + 1] and arreglo[high])
    return (i + 1)
}
```

3. Desarrollo

Suponga que los datos de los estudiantes de una escuela se almacenan en una estructura que tiene los siguientes campos:

- Nombre
- Carrera
- Promedio
- Créditos cursados

Utilizando un arreglo de 10 elementos la estructura mencionada, ordene en forma ascendente (a-z) los datos por nombre y elabore una tabla con los siguientes datos:
Para los pasos a-d utilice el pivote que prefiera.

a. El tiempo y la cantidad de iteraciones para el peor de los casos
b. El tiempo y la cantidad de iteraciones para el mejor de los casos
c. El tiempo y la cantidad de iteraciones para cualquier otro caso
d. ¿Qué pasa si todos los datos son iguales?

Utilizando el mismo conjunto de datos para los siguientes pasos determine los puntos e-h:

e. ¿Cantidad de pasadas si el pivote es el elemento de la izquierda?
f. ¿Cantidad de pasadas si el pivote es el elemento de la derecha?
g. ¿Cantidad de pasadas si el pivote es el elemento del centro del arreglo?
h. ¿Cantidad de pasadas si el pivote es un elemento aleatorio?

Repita los pasos e-h con otro conjunto de datos y responda.

- ¿La selección del pivote afectó de alguna manera la cantidad de pasadas que hace el algoritmo?
- ¿Hubo cambios al utilizar otros datos o se comportó igual que con el primero?
- ¿Cuál forma de elegir el pivote resultó más eficiente?

- No se pide que capture los datos, inicialice el arreglo con los valores necesarios.
- Elabore y suba a moodle un reporte en formato PDF con su código, los resultados obtenidos organizados en tablas y sus conclusiones.
- La implementación debe ser un trabajo original.
- La conclusión NO es la explicación de cómo funciona el algoritmo, historia del algoritmo, etc. sino una reflexión propia de los resultados obtenidos en los experimentos.
- La buena presentación y ortografía serán consideradas como parte de la evaluación.
- No se aceptarán trabajos plagiados, de otros compañeros o traducidos de otros lenguajes.

4. Resultados

4.1 Implementación en C

```c
/*Practica 10:
Elaborado por Reyes Udasco Rachelle Nerie*/

//Declaracion de bibliotecas
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

//Definicion de valores constantes
#define MAX 50   //Tamano de cadena maximo
#define registros 10  //Cantidad de registros

//Estructura del estudiante
typedef struct Estudiante_ {
    char nombre[MAX];
    char carrera[MAX];
    float promedio;
    int creditosCursados;
}Estudiante;

//Prototipo de funciones
void imprimirEstudiantes(Estudiante *arreglo, int n);
void imprimirNombres(Estudiante *arreglo, int n);
void swap(void* estudianteA, void* estudianteB, size_t tamano);
int particionDer (Estudiante *arreglo, int low, int high);
int particionIzq(Estudiante *arreglo, int low, int high);
int particionMed(Estudiante *arreglo, int low, int high);
int particionRand(Estudiante *arreglo, int low, int high);

void quickSort(Estudiante *arreglo, int low, int high,int *iteraciones);

//Funcion principal
int main(){
    int iteraciones = 0;
    double tiempo;
    clock_t inicio, fin;  //Variables utilizadas para medir el tiempo

    //Arreglo no ordenado
   Estudiante estudiante[registros] = {  //Declaracion e inicializacion
de la variable
        {"Cesena Rosa", "Docencia",98.50,200},
        {"Robles Karla", "Docencia",95.20,123},
        {"Lagarda Paola", "QFB",97.8,197},
        {"Galindo Itzel", "Economia",90.50,176},
        {"Soto Paola", "Odontologia",89.50,234},
        {"Leal Rodrigo", "Ing. en Computacion",87.50,157},
        {"Hernandez Alexia", "Mercadotecnia",92.75,134},
        {"Saucedo Sylvia", "Idiomas",92.34,165},
        {"Gamboa Israel", "Psicologia",89.50,236},
        {"Hernandez Edson", "Idiomas",90.58,189}
    };
/*
    //Arreglo ordenado
```

```
    Estudiante estudiante[registros] = {  //Declaracion e inicializacion
de la variable
        {"Cesena Rosa", "Docencia",98.50,200},
        {"Galindo Itzel", "Economia",90.50,176},
        {"Gamboa Israel", "Psicologia",89.50,236},
        {"Hernandez Alexia", "Mercadotecnia",92.75,134},
        {"Hernandez Edson", "Idiomas",90.58,189},
        {"Lagarda Paola", "QFB",97.8,197},
        {"Leal Rodrigo", "Ing. en Computacion",87.50,157},
        {"Robles Karla", "Docencia",95.20,123},
        {"Saucedo Sylvia", "Idiomas",92.34,165},
        {"Soto Paola", "Odontologia",89.50,234}
    };
*/
/*
 //Arreglo
   Estudiante estudiante[registros] = {  //Declaracion e inicializacion
de la variable
        {"Cesena Rosa", "Docencia",98.50,200},
        {"Robles Karla", "Docencia",95.20,123},
        {"Lagarda Paola", "QFB",97.8,197},
        {"Hernandez Alexia", "Mercadotecnia",92.75,134},
        {"Galindo Itzel", "Economia",90.50,176},
        {"Soto Paola", "Odontologia",89.50,234},
        {"Leal Rodrigo", "Ing. en Computacion",87.50,157},
        {"Hernandez Edson", "Idiomas",90.58,189},
        {"Saucedo Sylvia", "Idiomas",92.34,165},
        {"Gamboa Israel", "Psicologia",89.50,236}
    };
*/
/*
    //Arreglo de datos iguales
   Estudiante estudiante[registros] = {  //Declaracion e inicializacion
de la variable
        {"Robles Karla", "Docencia",95.20,123},
        {"Robles Karla", "Docencia",95.20,123},
        {"Robles Karla", "Docencia",95.20,123},
        {"Robles Karla", "Docencia",95.20,123},
        {"Robles Karla", "Docencia",95.20,123},
        {"Robles Karla", "Docencia",95.20,123},
        {"Robles Karla", "Docencia",95.20,123},
        {"Robles Karla", "Docencia",95.20,123},
        {"Robles Karla", "Docencia",95.20,123},
        {"Robles Karla", "Docencia",95.20,123},
    };
*/
/*
 //Arreglo con otros datos
   Estudiante estudiante[registros] = {  //Declaracion e inicializacion
de la variable
        {"Ruiz Olivia", "Ing. en Quimica",98.50,200},
        {"Torres Gustavo", "Ing. en Computacion",95.20,123},
        {"Angulo Ana", "Mercadotecnia",97.8,197},
        {"Osuna Eduardo", "Administracion",92.75,134},
        {"Agreda Samara", "Negocios Internacionales",90.50,176},
        {"Aguila Santiago", "Ing. en Sistemas",89.50,234},
        {"Castellon Alan", "Energias Renovables",87.50,157},
        {"Cota Brandon", "Ing. en Sistemas",90.58,189},
        {"Soto Alex", "Energias Renovables",92.34,165},
        {"Ruiz Carissa", "Medicina",89.50,236}
```

```
    };
*/
    printf(" Arreglo original\n");
    imprimirNombres(estudiante, 10);

    inicio = clock();  //Inicia el reloj
    quickSort(estudiante, 0, registros-1, &iteraciones);
    fin = clock();     //Detiene el reloj
    tiempo = (double)(fin - inicio)/CLOCKS_PER_SEC;  //Obtiene el tiempo
transucrrido

    printf("\n\n Iteraciones: %d", iteraciones);
    printf("\n Tiempo transcurrido en s: %f\n ", tiempo);

printf("\n\n%20s\t%20s\t%15s\t%15s","NOMBRE","CARRERA","PROMEDIO","CREDIT
OS");
    imprimirEstudiantes(estudiante,registros);

    return 0;
}

//Procedimiento que imprime los registros de estudiantes
void imprimirEstudiantes(Estudiante *arreglo, int n) {
    int i;

    for(i=0;i<n;i++) {

printf("\n%20s\t%20s\t%15f%15d",arreglo[i].nombre,arreglo[i].carrera,arre
glo[i].promedio,arreglo[i].creditosCursados);
    }
    printf("\n");
}

//Procedimiento que imprime los nombres
void imprimirNombres(Estudiante *arreglo, int n) {
    int i;

    for(i=0;i<n;i++) {
        printf(" %s,",arreglo[i].nombre);
        if(i==4)
            printf("\n");
    }
}

//Funcion que intercambia el valor de dos variables
void swap(void* estudianteA, void* estudianteB, size_t tamano) {
    void* tmp = malloc(tamano);
    memcpy(tmp, estudianteA, tamano);
    memcpy(estudianteA, estudianteB, tamano);
    memcpy(estudianteB, tmp, tamano);
    free(tmp);
}


//Funcion que realiza la particion con el elemento a la derecha
int particionDer (Estudiante *arreglo, int low, int high) {
    Estudiante auxiliar, pivote;
    pivote = arreglo[high];
    int i=0, j;
    i= (low-1);
```

```
    for (j=low; j<=high-1; j++) {
        if (strcmp(arreglo[j].nombre, pivote.nombre)<=0) {
            i++;
            swap(&arreglo[i],&arreglo[j],sizeof(Estudiante));
        }
    }
    swap(&arreglo[i+1],&arreglo[high],sizeof(Estudiante));
    return (i+1);
}

//Funcion que realiza la particion con el elemento a la izquierda
int particionIzq(Estudiante *arreglo, int low, int high) {

   Estudiante pivote = arreglo[low], temp;
   int j,i = low+1;

   for(j=low+1;j<=high;j++) {
    if(strcmp(arreglo[j].nombre, pivote.nombre)<0) {
        if(i!=j) {
            swap(&arreglo[i],&arreglo[j],sizeof(Estudiante));
        }
        i++;
    }
   }
    arreglo[low] = arreglo[i-1];
    arreglo[i-1] = pivote;
    return i-1;
}

//Funcion que realiza la particion con el elemento en el medio
int particionMed(Estudiante *arreglo, int low, int high) {

    int indicePivote = (low+high)/2;
    Estudiante pivote = arreglo[indicePivote];
    swap(&arreglo[indicePivote],&arreglo[high],sizeof(Estudiante));

    indicePivote = high;
    int i = low -1;

    for(int j=low; j<=high-1; j++) {
        if(strcmp(arreglo[j].nombre, pivote.nombre)<=0) {
            i = i+1;
            swap(&arreglo[i],&arreglo[j],sizeof(Estudiante));

        }
    }
    swap(&arreglo[i+1],&arreglo[indicePivote],sizeof(Estudiante));
    return i+1;

}

//Funcion que realiza la particion con un elemento al azar
int particionRand(Estudiante *arreglo, int low, int high) {

    srand(time(NULL));
    int indicePivote = low + rand() % (high-low+1);
    Estudiante pivote = arreglo[indicePivote];
    swap(&arreglo[indicePivote],&arreglo[high],sizeof(Estudiante));
```

```
    indicePivote = high;
    int i = low -1;

    for(int j=low; j<=high-1; j++) {
        if(strcmp(arreglo[j].nombre, pivote.nombre)<=0) {
            i = i+1;
            swap(&arreglo[i],&arreglo[j],sizeof(Estudiante));

        }
    }
    swap(&arreglo[i+1],&arreglo[indicePivote],sizeof(Estudiante));
    return i+1;

}

//Funcion que realiza el quickSort
void quickSort(Estudiante *arreglo, int low, int high, int * iteraciones)
{
    int pivote;
    if (low<high) {
        *iteraciones = *iteraciones +1;
        pivote = particionDer(arreglo,low,high);
        printf("\n\n Pivote: %d, Valor:
%s\n",pivote,arreglo[pivote].nombre);
        imprimirNombres(arreglo, registros);
        quickSort(arreglo, low, pivote-1, iteraciones);
        quickSort(arreglo, pivote+1, high, iteraciones);
    }
}

/* //Funcion que realiza el quickSort sin impresiones
void quickSort(Estudiante *arreglo, int low, int high) {
    int pivote;

    if (low<high) {
        pivote = particionDer(arreglo,low,high);
        quickSort(arreglo, low, pivote-1);
        quickSort(arreglo, pivote+1, high);
    }
}
*/
```

Impresión de la estructura completa

```
         NOMBRE                CARRERA          PROMEDIO        CREDITOS
     Cesena Rosa                Docencia       98.500000             200
   Galindo Itzel                Economia       90.500000             176
   Gamboa Israel              Psicologia       89.500000             236
Hernandez Alexia          Mercadotecnia       92.750000             134
 Hernandez Edson                Idiomas       90.580002             189
   Lagarda Paola                    QFB       97.800003             197
    Leal Rodrigo  Ing. en Computacion       87.500000             157
    Robles Karla                Docencia       95.199997             123
  Saucedo Sylvia                Idiomas       92.339996             165
      Soto Paola             Odontologia       89.500000             234
```

4.2 Tablas y análisis de resultados

Tabla 1.Tiempo e iteraciones de los diferentes casos

	Peor Caso	Mejor caso	Caso Promedio	Datos Iguales
Tiempo	0.031s	0.004s	0.008s	0.027
Iteraciones	9	6	6	9

Para los pasos a-d utilice el pivote que prefiera.

a. Peor caso O(log(log n))
Para este caso se utilizó un arreglo ordenado y un pivote se encuentra en uno de sus extremos. Como lo esperado, el número de iteraciones y el tiempo es mayor que los otros casos.

b. **Mejor caso Ω (n)**
Este caso se presenta cuando el arreglo no está ordenado y el pivote cae en el centro del arreglo. El tiempo de ejecución es significativamente menor que el de los otros casos, en el caso de las iteraciones, comparte el mismo resultado con el caso promedio.

c. Caso Promedio θ(log(log n))
Se presenta cuando el pivote se encuentra alrededor del centro de un arreglo no completamente ordenado. El tiempo y el número de iteraciones es aproximadamente igual el del mejor caso.

d. ¿Qué pasa si todos los datos son iguales?
De acuerdo al resultado obtenido, realiza la misma cantidad de iteraciones que el peor caso, esto significa que este método de ordenación realiza la misma cantidad de comparaciones sin importar si los valores son iguales.

Tabla 2.Cantidad de pasadas de los diferentes pivotes

Cantidad de Pasadas	Pivote Derecha	Pivote Izquierda	Pivote centro	Pivote Aleatorio
Primer arreglo	7	6	6	6
Segundo arreglo	6	6	7	6

- ¿La selección del pivote afectó de alguna manera la cantidad de pasadas que hace el algoritmo?
  En base a los datos adquiridos, la elección de un pivote con respecto a su posición no necesariamente afecta la cantidad de pasadas que realiza. La selección de un pivote eficiente depende directamente si este va a quedarse en el centro, es decir, en una posición en la cual la partición va a ser a la mitad.

- ¿Hubo cambios al utilizar otros datos o se comportó igual que con el primero?
  Tuvo un comportamiento relativamente igual que con el primero, la cantidad de pasadas resultó ser iguales en diferentes casos por la forma en que el arreglo se encuentra ordenado.

- ¿Cuál forma de elegir el pivote resultó más eficiente?
  Con los resultados obtenidos, no se podría concluir cuál es el más eficiente. Al elegir un pivote por su posición no nos garantiza si será eficiente. Por ejemplo, en caso de que el pivote es fijo y que resulta ser el elemento más pequeño o más grande, una de las particiones quedara vacía. Aunque no es la mejor solución, el generar un pivote aleatorio disminuye las posibilidades de elegir uno de los dos datos.

4.3 Anexo capturas de pantalla

a) Peor de los casos

```
Arreglo original
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 9, Valor: Soto Paola
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 8, Valor: Saucedo Sylvia
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 7, Valor: Robles Karla
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 6, Valor: Leal Rodrigo
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 5, Valor: Lagarda Paola
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 4, Valor: Hernandez Edson
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 3, Valor: Hernandez Alexia
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 2, Valor: Gamboa Israel
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 1, Valor: Galindo Itzel
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Iteraciones: 9
Tiempo transcurrido en s: 0.031000
```

b) Mejor de los casos

```
Arreglo original
Robles Karla, Lagarda Paola, Galindo Itzel, Soto Paola, Leal Rodrigo,
Hernandez Alexia, Cesena Rosa, Saucedo Sylvia, Gamboa Israel, Hernandez Edson,

Pivote: 4, Valor: Hernandez Edson
Galindo Itzel, Hernandez Alexia, Cesena Rosa, Gamboa Israel, Hernandez Edson,
Lagarda Paola, Robles Karla, Saucedo Sylvia, Soto Paola, Leal Rodrigo,

Pivote: 2, Valor: Gamboa Israel
Galindo Itzel, Cesena Rosa, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Robles Karla, Saucedo Sylvia, Soto Paola, Leal Rodrigo,

Pivote: 0, Valor: Cesena Rosa
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Robles Karla, Saucedo Sylvia, Soto Paola, Leal Rodrigo,

Pivote: 6, Valor: Leal Rodrigo
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Saucedo Sylvia, Soto Paola, Robles Karla,

Pivote: 7, Valor: Robles Karla
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Soto Paola, Saucedo Sylvia,

Pivote: 8, Valor: Saucedo Sylvia
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Iteraciones: 6
Tiempo transcurrido en s: 0.004000
```

c) Caso promedio

```
Arreglo original
Cesena Rosa, Robles Karla, Lagarda Paola, Galindo Itzel, Soto Paola,
Leal Rodrigo, Hernandez Alexia, Saucedo Sylvia, Gamboa Israel, Hernandez Edson,

Pivote: 4, Valor: Hernandez Edson
Cesena Rosa, Galindo Itzel, Hernandez Alexia, Gamboa Israel, Hernandez Edson,
Leal Rodrigo, Lagarda Paola, Saucedo Sylvia, Robles Karla, Soto Paola,

Pivote: 2, Valor: Gamboa Israel
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Leal Rodrigo, Lagarda Paola, Saucedo Sylvia, Robles Karla, Soto Paola,

Pivote: 1, Valor: Galindo Itzel
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Leal Rodrigo, Lagarda Paola, Saucedo Sylvia, Robles Karla, Soto Paola,

Pivote: 9, Valor: Soto Paola
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Leal Rodrigo, Lagarda Paola, Saucedo Sylvia, Robles Karla, Soto Paola,

Pivote: 7, Valor: Robles Karla
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Leal Rodrigo, Lagarda Paola, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 5, Valor: Lagarda Paola
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Iteraciones: 6
Tiempo transcurrido en s: 0.008000
```

d) Datos iguales

```
Arreglo original
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,

Pivote: 9, Valor: Robles Karla
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,

Pivote: 8, Valor: Robles Karla
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,

Pivote: 7, Valor: Robles Karla
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,

Pivote: 6, Valor: Robles Karla
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,

Pivote: 5, Valor: Robles Karla
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,

Pivote: 4, Valor: Robles Karla
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,

Pivote: 3, Valor: Robles Karla
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,

Pivote: 2, Valor: Robles Karla
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,

Pivote: 1, Valor: Robles Karla
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,
Robles Karla, Robles Karla, Robles Karla, Robles Karla, Robles Karla,

Iteraciones: 9
Tiempo transcurrido en s: 0.027000
```

e) Pivote elemento de la izquierda

```
Arreglo original
Cesena Rosa, Robles Karla, Lagarda Paola, Hernandez Alexia, Galindo Itzel,
Soto Paola, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Gamboa Israel,

Pivote: 0, Valor: Cesena Rosa
Cesena Rosa, Robles Karla, Lagarda Paola, Hernandez Alexia, Galindo Itzel,
Soto Paola, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Gamboa Israel,

Pivote: 7, Valor: Robles Karla
Cesena Rosa, Gamboa Israel, Lagarda Paola, Hernandez Alexia, Galindo Itzel,
Leal Rodrigo, Hernandez Edson, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 2, Valor: Gamboa Israel
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Lagarda Paola,
Leal Rodrigo, Hernandez Edson, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 3, Valor: Hernandez Alexia
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Lagarda Paola,
Leal Rodrigo, Hernandez Edson, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 5, Valor: Lagarda Paola
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 8, Valor: Saucedo Sylvia
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Iteraciones: 6
Tiempo transcurrido en s: 0.015000
```

Arreglo con otros datos

```
Arreglo original
Ruiz Olivia, Torres Gustavo, Angulo Ana, Osuna Eduardo, Agreda Samara,
Aguila Santiago, Castellon Alan, Cota Brandon, Soto Alex, Ruiz Carissa,

Pivote: 7, Valor: Ruiz Olivia
Ruiz Carissa, Angulo Ana, Osuna Eduardo, Agreda Samara, Aguila Santiago,
Castellon Alan, Cota Brandon, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 6, Valor: Ruiz Carissa
Cota Brandon, Angulo Ana, Osuna Eduardo, Agreda Samara, Aguila Santiago,
Castellon Alan, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 4, Valor: Cota Brandon
Castellon Alan, Angulo Ana, Agreda Samara, Aguila Santiago, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 3, Valor: Castellon Alan
Aguila Santiago, Angulo Ana, Agreda Samara, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 1, Valor: Aguila Santiago
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 8, Valor: Soto Alex
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Iteraciones: 6
Tiempo transcurrido en s: 0.004000
```

f) Pivote elemento de la derecha

```
Arreglo original
Cesena Rosa, Robles Karla, Lagarda Paola, Hernandez Alexia, Galindo Itzel,
Soto Paola, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Gamboa Israel,

Pivote: 2, Valor: Gamboa Israel
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Robles Karla,
Soto Paola, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Lagarda Paola,

Pivote: 1, Valor: Galindo Itzel
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Robles Karla,
Soto Paola, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Lagarda Paola,

Pivote: 5, Valor: Lagarda Paola
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 4, Valor: Hernandez Edson
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 9, Valor: Soto Paola
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 8, Valor: Saucedo Sylvia
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 7, Valor: Robles Karla
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Iteraciones: 7
Tiempo transcurrido en s: 0.008000
```

Arreglo con otros datos

```
Arreglo original
Ruiz Olivia, Torres Gustavo, Angulo Ana, Osuna Eduardo, Agreda Samara,
Aguila Santiago, Castellon Alan, Cota Brandon, Soto Alex, Ruiz Carissa,

Pivote: 6, Valor: Ruiz Carissa
Angulo Ana, Osuna Eduardo, Agreda Samara, Aguila Santiago, Castellon Alan,
Cota Brandon, Ruiz Carissa, Torres Gustavo, Soto Alex, Ruiz Olivia,

Pivote: 4, Valor: Cota Brandon
Angulo Ana, Agreda Samara, Aguila Santiago, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Torres Gustavo, Soto Alex, Ruiz Olivia,

Pivote: 3, Valor: Castellon Alan
Angulo Ana, Agreda Samara, Aguila Santiago, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Torres Gustavo, Soto Alex, Ruiz Olivia,

Pivote: 1, Valor: Aguila Santiago
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Torres Gustavo, Soto Alex, Ruiz Olivia,

Pivote: 7, Valor: Ruiz Olivia
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 9, Valor: Torres Gustavo
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Iteraciones: 6
Tiempo transcurrido en s: 0.003000
```

g) Pivote elemento del centro

```
Arreglo original
Cesena Rosa, Robles Karla, Lagarda Paola, Hernandez Alexia, Galindo Itzel,
Soto Paola, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Gamboa Israel,

Pivote: 1, Valor: Galindo Itzel
Cesena Rosa, Galindo Itzel, Lagarda Paola, Hernandez Alexia, Gamboa Israel,
Soto Paola, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Robles Karla,

Pivote: 9, Valor: Soto Paola
Cesena Rosa, Galindo Itzel, Lagarda Paola, Hernandez Alexia, Gamboa Israel,
Robles Karla, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Soto Paola,

Pivote: 7, Valor: Robles Karla
Cesena Rosa, Galindo Itzel, Lagarda Paola, Hernandez Alexia, Gamboa Israel,
Leal Rodrigo, Hernandez Edson, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 2, Valor: Gamboa Israel
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Leal Rodrigo, Lagarda Paola, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 4, Valor: Hernandez Edson
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Leal Rodrigo, Lagarda Paola, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 6, Valor: Leal Rodrigo
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Iteraciones: 6
Tiempo transcurrido en s: 0.028000
```

Arreglo con otros datos

```
Arreglo original
Ruiz Olivia, Torres Gustavo, Angulo Ana, Osuna Eduardo, Agreda Samara,
Aguila Santiago, Castellon Alan, Cota Brandon, Soto Alex, Ruiz Carissa,

Pivote: 0, Valor: Agreda Samara
Agreda Samara, Torres Gustavo, Angulo Ana, Osuna Eduardo, Ruiz Carissa,
Aguila Santiago, Castellon Alan, Cota Brandon, Soto Alex, Ruiz Olivia,

Pivote: 1, Valor: Aguila Santiago
Agreda Samara, Aguila Santiago, Angulo Ana, Osuna Eduardo, Ruiz Carissa,
Ruiz Olivia, Castellon Alan, Cota Brandon, Soto Alex, Torres Gustavo,

Pivote: 7, Valor: Ruiz Olivia
Agreda Samara, Aguila Santiago, Angulo Ana, Osuna Eduardo, Ruiz Carissa,
Castellon Alan, Cota Brandon, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 6, Valor: Ruiz Carissa
Agreda Samara, Aguila Santiago, Angulo Ana, Osuna Eduardo, Cota Brandon,
Castellon Alan, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 5, Valor: Osuna Eduardo
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 3, Valor: Castellon Alan
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 8, Valor: Soto Alex
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Iteraciones: 7
Tiempo transcurrido en s: 0.027000
```

h) Pivote elemento aleatorio

```
Arreglo original
Cesena Rosa, Robles Karla, Lagarda Paola, Hernandez Alexia, Galindo Itzel,
Soto Paola, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Gamboa Israel,

Pivote: 2, Valor: Gamboa Israel
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Robles Karla,
Soto Paola, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Lagarda Paola,

Pivote: 1, Valor: Galindo Itzel
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Robles Karla,
Soto Paola, Leal Rodrigo, Hernandez Edson, Saucedo Sylvia, Lagarda Paola,

Pivote: 5, Valor: Lagarda Paola
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 4, Valor: Hernandez Edson
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 7, Valor: Robles Karla
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Pivote: 9, Valor: Soto Paola
Cesena Rosa, Galindo Itzel, Gamboa Israel, Hernandez Alexia, Hernandez Edson,
Lagarda Paola, Leal Rodrigo, Robles Karla, Saucedo Sylvia, Soto Paola,

Iteraciones: 6
Tiempo transcurrido en s: 0.007000
```

Arreglo con otros datos

```
Arreglo original
Ruiz Olivia, Torres Gustavo, Angulo Ana, Osuna Eduardo, Agreda Samara,
Aguila Santiago, Castellon Alan, Cota Brandon, Soto Alex, Ruiz Carissa,

Pivote: 1, Valor: Aguila Santiago
Agreda Samara, Aguila Santiago, Angulo Ana, Osuna Eduardo, Ruiz Olivia,
Ruiz Carissa, Castellon Alan, Cota Brandon, Soto Alex, Torres Gustavo,

Pivote: 6, Valor: Ruiz Carissa
Agreda Samara, Aguila Santiago, Angulo Ana, Osuna Eduardo, Castellon Alan,
Cota Brandon, Ruiz Carissa, Torres Gustavo, Soto Alex, Ruiz Olivia,

Pivote: 4, Valor: Cota Brandon
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Torres Gustavo, Soto Alex, Ruiz Olivia,

Pivote: 3, Valor: Castellon Alan
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Torres Gustavo, Soto Alex, Ruiz Olivia,

Pivote: 7, Valor: Ruiz Olivia
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Pivote: 9, Valor: Torres Gustavo
Agreda Samara, Aguila Santiago, Angulo Ana, Castellon Alan, Cota Brandon,
Osuna Eduardo, Ruiz Carissa, Ruiz Olivia, Soto Alex, Torres Gustavo,

Iteraciones: 6
Tiempo transcurrido en s: 0.016000
```

5. Conclusiones

El método Quicksort, como su nombre lo indica, parece ser uno de los algoritmos de ordenación que ofrece mayor velocidad con respecto a su tiempo de ejecución. El algoritmo es relativamente fácil de comprender e implementar aunque es largo o utiliza más de una función, dependiendo de cómo fue programada. En caso de esta práctica, se utilizaron tres funciones principales, una para las particiones, otra para el intercambio de valores y otra que realiza las llamadas recursivas de la función Quicksort.

Al realizar el análisis empírico para diferentes casos puedo concluir que el orden de los elementos del arreglo y el valor del pivote son muy importantes al momento de realizar las particiones. Sin embargo, cabe destacar que el querer seleccionar el pivote que puede resultar más eficiente no siempre es viable. Este se debe a que si en el programa se establece de manera fija la posición del pivote, en cada iteración se estaría eligiendo un valor al azar que puede caer en cualquier posición generando una partición inequitativa en el arreglo y por ende un mayor tiempo de ejecución.