Informe Prática 01

Análisis de los algoritmos de ordenamiento InsertionSort y QuickSort, y el algoritmo de búsqueda binaria

Nota

Estudiantes	Escuela	Asignatura
Arias Quispe, Jhonatan David Mamani Huarsaya, Jorge Luis Mollo Chuquicaña, Dolly Yadhira Quispe Condori, Alvaro Raul Velarde Saldaña, Jhossep Fabritzio jariasq@unsa.edu.pe jmamanihuars@unsa.edu.pe dmolloc@unsa.edu.pe aquispecondo@unsa.edu.pe jvelardesa@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Fundamentos de Programación 2 Semestre: II Código: 17013

Teoría	Tema	Duración
Prática 01	Análisis de los algoritmos de ordenamiento InsertionSort y QuickSort, y el algoritmo de búsqueda binaria	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 25 Setiembre 2023	Al 4 octubre 2023

1. Actividad

- Elaborar un proyecto utilizando git. donde se elabore un sistema para ingresar datos de alumnos universitarios. (Clase Student)
- El sistema debe almacenar los estudiantes en un Array. (Considerar leer archivos CSV).
- Implemente el algoritmo de ordenamiento por Inserción(Iterativo-Cuadrático) para ordenar el arreglo de estudiantes por diferentes parámetros. Ejemplo: Por apellido, paterno. Descubra cuál es el tiempo que se demora en las ejecuciones.
- Explique cualquier otro algoritmo de ordenamiento de complejidad logarítmica. e implemente el ordenamiento utilizando los mismo parámetros anteriores.

- Grafique los resultados de las simulaciones realizadas considerando como unidad de medida los nanosegundos. Desde n=1 alumno hasta n=N alumnos.
- Luego, para el arreglo ordenado implemente el algoritmo de búsqueda binaria iterativo/recursivo y grafique los resultados de sus simulaciones.

2. Equipos, materiales y temas utilizados

- Sistema Operativo ArchCraft GNU Linux 64 bits Kernell, Sistema Operativo Ubuntu GNU Linux 22.04, Sistema operativo windows 11
- NeoVim, vs code
- OpenJDK 64-Bit 20.0.1
- Gnuplot 5.4.9
- Git 2.42.0
- Cuenta en GitHub con el correo institucional.
- Programación Orientada a Objetos.
- Algoritmos de ordenamiento y búsqueda.

3. URL de Repositorio Github

- URL para acceder a la Prática 01 en el Repositorio GitHub.
- `https://github.com/JhonatanDczel/prac01`

4. Documentacion

5. Desarrollo de la actividad

5.1. Lectura y almacenamiento de datos

5.1.1. Descripción

La clase `Reader` se encarga de leer los datos de un archivo CSV y almacenarlos en un arreglo de objetos `Student`. Los pasos principales son:

- Abrir el archivo `data.csv`
- Leer linea por linea con `BufferedReader`
- Separar cada linea por comas para obtener los campos
- Crear un nuevo objeto `Student`
- Setear los campos leídos en el objeto `Student`
- Agregar el objeto `Student` a un `ArrayList`
- Convertir el `ArrayList` a un arreglo al final

5.1.2. Código relevante

Listing 1: Método para leer archivo CSV

```java
private void readDataFile(){

  BufferedReader reader = null;

  String line = "";

  String[] parts;

  try {

    FileReader fileReader = new FileReader("./reader/data.csv");

    reader = new BufferedReader(fileReader);

    while ((line = reader.readLine()) != null) {

      parts = line.split(",");

      //... creacin y seteo de Student

    }

  }catch (Exception e){

    e.printStackTrace();

  }finally{

    //...cerrar reader

  }

}
```

Listing 2: Conversión de ArrayList a arreglo

```java
classmates = students.toArray(new Student[students.size()]);
```

5.1.3. Explicación

Se utiliza `BufferedReader` para leer el archivo línea por línea de forma eficiente. Cada línea se divide en las partes correspondientes a cada campo, delimitadas por comas. Luego se crea un objeto `Student`, se setean sus campos y se agrega al `ArrayList`. Al final se convierte el `ArrayList` a arreglo para retornarlo.

5.2. Algoritmo de insercion

El ordenamiento por inserción es un algoritmo de ordenamiento simple y eficiente que funciona de la siguiente manera:

- Comienza con un arreglo desordenado.
- Divide el arreglo en dos partes: una parte ordenada y una parte desordenada.
- En cada iteración, toma el primer elemento de la parte desordenada y lo compara con los elementos en la parte ordenada.

- Inserta el elemento en la posición correcta en la parte ordenada, desplazando los elementos mayores a la derecha.
- Repite este proceso hasta que la parte desordenada esté vacía y todos los elementos estén en la parte ordenada.

```
INSERTION-SORT(A, n)
1  for i = 2 to n
2      key = A[i]
3      // Insert A[i] into the sorted subarray A[1 : i − 1].
4      j = i − 1
5      while j > 0 and A[j] > key
6          A[j + 1] = A[j]
7          j = j − 1
8      A[j + 1] = key
```

Figura 1: Pseudocódigo del ordenamiento por inserción

El ordenamiento por inserción es eficiente para arreglos pequeños o parcialmente ordenados, pero puede volverse ineficiente para arreglos grandes debido a su complejidad cuadrática en el peor de los casos. Sin embargo, es estable y fácil de implementar.

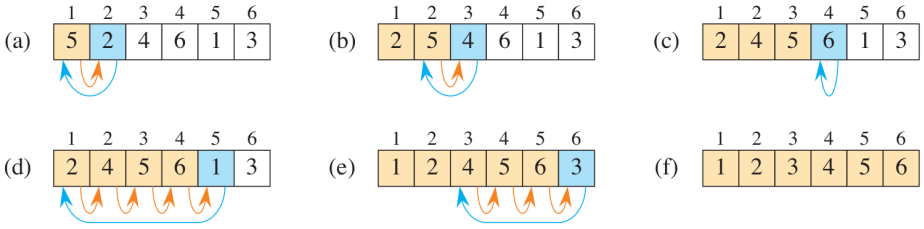

Figura 2: Iteraciones del ordenamiento por inserción

Entendido el funcionamiento de este algoritmo, podemos aplicarlo para resolver esta actividad.

- Comprendamos que nosotros estamos trabajando con un arreglo de objetos, es decir, trabajamos con referencias a objetos que se estan almacenando en cada posición del arreglo.
- Las comparaciones se hacen según los atributos de la clase Student, por lo tanto, tenemos la necesidad de usar sus métodos accesores; en este caso solo usamos los métodos getters.
- La claridad es mejor que la astucia.

5.2.1. Commits principales

- **Hash: 971ab17de7c8390a9fee07b569b37cbc68cfed6f**
- En el presente commit se implementó el algorítmo de ordenamiento de Inserción donde nos aventuramos a diseñar el algoritmo usando nuestra propia lógica. Sim embargo, más adelante encontraremos formas mas eficientes del algoritmo.

Listing 3: Commit: Se implementó Quicksort para cui

```
  public static class InsertionAlgorithmCui (Student[] compaeritos ) {
      int lowest;
      Student pivot;
for(int i = 0; i < compaeritos.length; i++){
  lowest = i;
  pivot = compaeritos[i];
  while((lowest > 0) && (compaeritos[lowest - 1].getCui() > pivot.getCui())){
    compaeritos[lowest] = compaeritos[lowest - 1];
    lowest--;
  }
   compaeritos[lowest] = pivot;
    }
   }
```

- **Hash: 4fe74101fa064fa87663c6f674902add0afde893**
- En el presente commit se realizo la elaboración de un metodo privado para comparar Strings haci como se realizo unas modificación al metodo compareEmail() que presento algunos errores anteriormente.

Listing 4: Commit: Implementando los métodos que usan el algoritmo de inserción para ordenar según nombre y apellidos. Tambien se ha creado un método privado necesario para hacer las comparaciones de los Strings. Por ultimo se corrigió algunos errores de sintaxis y pequeños cambios para mayor comprensión y eficiencia

```

  private static boolean compareString(String word1, String word2){
    String lower1 = word1.toLowerCase();
    String lower2 = word1.toLowerCase();
    int length = lower1.length();
    if(length > lower2.length()){
      length = lower2.length();
    }
    int i = 0;
    while(i < length){
      if(lower1.charAt(i) == lower2.charAt(i)){
i++;
    }else{
if(lower1.charAt(i) > lower2.charAt(i)){
  return true;
}
return false;
    }
  }
  return false;
}
```

- **Hash: cda57aa68100e67a2c9f5ac809c67d11b231c28d**
- En el presente commit se muestra como se realizo un cambio en el argumento que reciben los metodos , esto debido a el avance realizado en la clase Reader. Se hicieron algunas mejoras en cuanto a la optimización de los metodos. Aunque más adelante encontraremos una forma aún más efectiva para realizar las comparaciones

Listing 5: Commit: Se cambio el argumento de todos los métodos implementados, de arreglo de Student a arreglo de Reader.Student para posteriormente realizar las pruebas. Por último en colaboracion con las pruebas se corrigieron algunos errores de sintaxis y se eliminó el método convertion porque si el atributo DateOfBirth se compara convirtiendolo a entero, no cumple para todos los casos.

```
1
2  private static boolean compareString(String word1, String word2){
3      //Suponemos que los datos no tienen mayusculas donde no deben. Adems este mtodo nos
4      //sirve para ordenar segn nacimiento, pues si convertimos a nmero, no cumple todos los casos
5
6    int length = word1.length();
7        if(length > word2.length()){
8          length = word2.length();
9        }
10       int i = 0;
11       while(i < length){
12         if(word1.charAt(i) == word2.charAt(i)){
13       i++;
14     }else{
15       if(word1.charAt(i) > word2.charAt(i)){
16         return true;
17       }
18     return false;
19         }
20   }
21   return false;
22 }
```

- **Hash: 99989865149d77dedee1467670ea5a45529154c4**
- En el presente commit encontramos un método mas corto y efectivo de realizar las comparaciones entre strings.Tambien se corrigieron los metodos de Gender y Status a los nuevos parámetros asignados del documento csv. Este metodo se puede implementar para varios parametros, cosa que notaremos más adelante

Listing 6: Commit: Algunas correcciones

```
1
2  private static boolean compareString(String word1, String word2){
3      word1 = word1.toUpperCase();
4      word2 = word2.toUpperCase();
5      if (word1.compareTo(word2) <= 0) return false;
6      return true;
7    }
```

- **Hash: 4da70da9049f884c22478d9aa71fcf9becbbc243**
- En el presente commit encontramos la aplicación del metodo compareString para todos los metodos, esto despues de darnos cuenta que este metodo funcionaria tambien para los otros campos . Se corrigio y cambio el algoritmo de Inserción iterativa cuadratica. En esta version aun se encuentra un error en el algoritmo el ciclo While de los algoritmos el cual no ordenaba el primer elemento, este error se corrigio en la version final del algoritmo

Listing 7: Commit: Nos aventuramos en una exploración cognitiva en la implementación del algoritmo Inserción, sin embargo regresamos al camino correcto donde se logra apreciar mejor su eficiencia y funcionalidad

```
1
2    public static void email(Reader.Student[] classmates){
3     int j;
4     Reader.Student pivot;
5     for(int i = 0; i < classmates.length; i++){
```

```
    j = i - 1;
    pivot = classmates[i];
    while(j > 0 && compareString(classmates[j].getEmail(), pivot.getEmail())){
      classmates[j + 1] = classmates[j];
          j--;
       }
       classmates[j + 1] = pivot;
    }
  }
```

- Hash:
-

Listing 8: TITULOOOOOO

5.3. Quicksort

- El método «Quicksort», también conocido como ordenación rápida, es uno de los algoritmos de ordenación más eficientes y ampliamente utilizados en la informática.
- Se basa en el paradigma de divíde y vencerás. Consiste en seleccionar un elemento de la lista, llamado "pivote,ƶ reorganizar los elementos en la lista de manera que los elementos menores que el pivote estén a su izquierda, y los elementos mayores estén a su derecha.

- Cuando se elige un pivote adecuado, «Quicksort» puede alcanzar su rendimiento óptimo, que es O(n log n) en promedio. Sin embargo, en el peor de los casos, se puede degradar a O(n ^{2}).

5.3.1. Commits de la implementación de Quicksort

- **Hash: afde7b901d3ac3c8414a3d6976fde97bcbe9729d**
- En el presente commit se implementó el quicksort para ordenar los CUI, sin embargo cuenta con un error. Cambia la referencia del atributo cui de la clase Student, sin embargo, no cambia la la referencia del objeto mismo, es un error que se solucionará más adelante.

Listing 9: Commit: Se implementó Quicksort para cui

```
public static void Cui (Reader.Student[] s, int left, int right) {
    int i = left;
    int j = right;
    int aux;
    while (i < j) {
        while (s[i].getCui() <= piv && i < j) i++;
        while (s[j].getCui() > piv) j--;
        if (i < j) {
            aux = s[i].getCui();
            s[i].setCui(s[j].getCui());
            s[j].setCui(aux);
        }
     }
    s[left].setCui(s[j].getCui());
    s[j].setCui(piv);
    if (left < j - 1) Cui(s, left, j -1);
    if (j + 1< right) Cui(s, j + 1, right);

 }
```

- **Hash: c9e70896b8edb82f963375cac9a4cb72d67ed6fb**
- En el presente commit de manera similar se implementó el método email para ordenar según la primera letra a los correos, que están representados mediante Strings, sin embargo, presenta el error de solo ordenar la primera letra. Más adelante se parchará el error.

Listing 10: Commit: Se añadió el método email para ordenar los correos según quicksort

```
1
2    public static void email (Reader.Student[] s, int left, int right) {
3        char piv = s[left].getEmail().charAt(0);
4        int i = left;
5        int j = right;
6        String aux;
7        while (i < j) {
8            while (s[i].getEmail().charAt(0) <= piv && i < j) i++;
9            while (s[j].getEmail().charAt(0) > piv) j--;
10           if (i < j) {
11               aux = s[i].getEmail();
12               s[i].setEmail(s[j].getEmail());
13               s[j].setEmail(aux);
14           }
15       }
16       aux = s[left].getEmail();
17       s[left].setEmail(s[j].getEmail());
18       s[j].setEmail(aux);
19       if (left < j - 1) email(s, left, j -1);
20       if (j + 1< right) email(s, j + 1, right);
21    }
```

- **Hash: cb4418c7361dc0e2da32042698b2400e79beebfd**
- Se creó la fución «smallerThan()» en la cual se usa el método String.compareTO(String), la función de esta misma es restar los caracteres según su orden ascii. Si w1 - w2 es positivo, significa que w1 debería estar después de w2.
- Esta función se utiliza cada vez que se quiere comparar dos String, solucionando el error mencionado anteriormente.

Listing 11: Commit: Se mejoró los métodos que ordenan Strings

```
1    public static boolean smallerThan(String w1, String w2) {
2        w1.toUpperCase();
3        w2.toUpperCase();
4        if (w1.compareTo(w2) <= 0) return true;
5        return false;
6
7    }
8
9  // Uso de smallerThan en otras funciones:
10
11   public static void lastNameM (Reader.Student[] s, int left, int right) {
12       String piv = s[left].getLastNameM();
13       int i = left;
14       int j = right;
15       String aux;
16       while (i < j) {
17           while (smallerThan(s[i].getLastNameM(), piv) && i < j) i++;
18           while (!smallerThan(s[j].getLastNameM(), piv)) j--;
19           if (i < j) {
20               aux = s[i].getLastNameM();
```

```
            s[i].setLastNameM(s[j].getLastNameM());
            s[j].setLastNameM(aux);
        }
    }
    aux = s[left].getLastNameM();
    s[left].setLastNameM(s[j].getLastNameM());
    s[j].setLastNameM(aux);
    if (left < j - 1) lastNameM (s, left, j -1);
    if (j + 1< right) lastNameM(s, j + 1, right);
}
```

- **Hash: 0c89058320e8f2dc1981a4666c3deb86df450702**
- Se solucioó el primer error mencionado, en esta oportunidad, el código intercambia la referencia de los objetos en el array en vez de sus atributos.

Listing 12: Commit: Se solucionó error de cambio de valores de atributos en vez de las referencias de los objetos

```
public static void email (Reader.Student[] s, int left, int right) {
    String piv = s[left].getEmail();
    int i = left;
    int j = right;
    Reader.Student aux;
    while (i < j) {
        while (smallerThan(s[i].getEmail(), piv) && i < j) i++;
        while (!smallerThan(s[j].getEmail(), piv)) j--;
        if (i < j) {
            aux = s[i];
            s[i] = s[j];
            s[j] = aux;
        }
    }
    aux = s[left];
    s[left] = s[j];
    s[j] = (aux);
    if (left < j - 1) email(s, left, j -1);
    if (j + 1< right) email(s, j + 1, right);
 }
```

5.4. Busqueda Binaria

- La búsqueda binaria es un algoritmo de búsqueda eficiente utilizado para encontrar un elemento específico en una lista ordenada.
- Tanto la versión recursiva como la iterativa de la búsqueda binaria se basan en el mismo principio: dividir y vencerás.
- En la búsqueda binaria recursiva, se divide repetidamente la lista en dos mitades y se compara el elemento buscado con el elemento en el medio. Este proceso se repite de manera recursiva hasta que se encuentre el elemento deseado
- La búsqueda binaria iterativa se implementa mediante un bucle en lugar de una función recursiva.

5.4.1. Forma iterativa

- **Hash: 55e501872e486ef65d414a41aeb9f673c6b8318e**

- Se creo «IterativeBinarySearch.java» donde se implementó el método cui() para realizar la búsqueda binaria de manera iterativa.

Listing 13: Commit: Se creó el método cui para la búsqueda binaria

```java
public static int cui (Reader.Student [] s, int x) {
  int l, r;
  l = 0;
  r = s.length - 1;
  while (l <= r) {
    int m = l + ( r - l ) / 2;
    if (s[m].getCui() == x) return m;
    if (s[m].getCui() < x) l = m + 1;
    else r = m - 1;
  }
  return -1;
}
```

- **Hash: b90e4fbe4c4373de351edf016e255e9534088251**
- De manera similar, se implementó el método « email() » usando como comparador la función «smallerThan()».
- Usando esta última función sabemos si la palabra buscada debería estar al delante o atrás.
- Usando este método que funciona con Strings, también se implementó el resto de funciones según los atributos de la clase Student.

Listing 14: Commit: Implementación para busqueda según email

```java
public static int email (Reader.Student [] s, String x) {
  int l = 0, r = s.length - 1;
  while (l <= r) {
    int m = l + (r - l) / 2;
    if (s[m].getEmail().equals(x)) return m;
    if (smallerThan(s[m].getEmail(), x)) l = m + 1;
    else r = m - 1;
  }
  return -1;
}

public static boolean smallerThan(String w1, String w2) {
    w1.toUpperCase();
    w2.toUpperCase();
    if (w1.compareTo(w2) <= 0) return true;
    return false;

  }

}
```

5.4.2. Forma recursiva

- **Hash: 2318dc5647d46b889c72f61a09a9fd7b8f8f2f5a**
- Posteriormente se creo el archivo «RecursiveBinarySearch.java» donde se implementaron las siguientes clases: «cui()» y «email()».

- Estas clases usan el algoritmo de la búsqueda binaria recursiva. Se reutiliza la función «smallerThan()» para comparar Strings.
- De manera análoga se implementa al resto de atributos de la clase Student que están basados en Strings, los cuales no se muestran en el presente informe para evitar una extensión innecesaria del mismo.

Listing 15: Commit: Se creó la búsqueda binaria con los métodos de cui y email

```
package algorithms;

import reader.Reader;

public class RecursiveBinarySearch {
  public static int cui (Reader.Student [] s, int x, int l, int r) {
    if (r >= l) {
      int m = l + (r - l) / 2;
      if (s[m].getCui() == x) return m;
      if (s[m].getCui() > x) return cui(s, x, l, m -1);
      return cui(s, x, m + 1, r);
    }
    return -1;
  }
  public static int email (Reader.Student [] s, String x, int l, int r) {
    if (r >= l) {
      int m = l + (r - l) / 2;
      if (s[m].getEmail().equals(x)) return m;
      if ( smallerThan(s[m].getEmail(), x) ) return email(s, x, m + 1, r);
      return email(s, x, l, m - 1);
    }
    return -1;
  }
  public static boolean smallerThan(String w1, String w2) {
      w1.toUpperCase();
      w2.toUpperCase();
      if (w1.compareTo(w2) <= 0) return true;
      return false;

    }

}
```

5.5. Parte grafica

Necesitamos una graficadora, que interprete el comportamiento de los algoritmos de ordenamiento, en un tiempo y muestra determinados...

- Para hacerlo usaremos GNUPlot
- Necesitaremos un Script que muestre por pantalla la grafica de los algoritmos

5.5.1. Script de bash para GNUPlot

Listing 16: Script de GNUPlot

```

#Definimos el nombre por defecto en caso de no haber ingresado un nombre en el segundo parametro
if [ "$outputFile" == ".png" ]; then
  outputFile="Grafica.png"
fi
```

```

#Ahora definimos los nombres de los archivos
grafico1=$(echo "$2" | cut -c 18- | rev | cut -c 5- | rev)
grafico2=$(echo "$3" | cut -c 18- | rev | cut -c 5- | rev)
#Definimos la linea que se usara para la salida del plot
plotLine="plot \"$2\" w lp lw 3 lc rgb \"#62aeef\" pt 6 title \"$grafico1\""
if [[ $3 != "" ]]; then
  plotLine="$plotLine, \"$3\" w lp lw 3 lc rgb \"#ddf056\" pt 6 t \"$grafico2\""
fi

# aqui se ejecuta Gnuplot para generar el grfico y exportarlo
gnuplot <<-EOF
  # Cambiamos el output de la terminal a png
  set terminal pngcairo enhanced
  set output "./graphics/imgs/$outputFile"

  set title "$1" textcolor rgb "white" font "helvetica,12"
  set key left box textcolor "white" font "helvetica,12"
  set object 1 rectangle from screen 0,0 to screen 1,1 behind noclip fc rgb "#32363d" fillstyle solid
       1.0
  set border lc rgb "white" lw 2
  set xlabel "Data count" font "helvetica,12" textcolor "white"
  set ylabel "Time (ns)" font "helvetica,12" textcolor "white"
  set tics font "helvetica,11" textcolor rgb "#faf95d"

  # Se crea la grafica
  $plotLine
EOF

xdg-open "./graphics/imgs/$outputFile"
echo "El grfico se ha generado como $outputFile"
```

- Este script recibe dos parametros:
- Nombre para la imagen del resultado
- Archivo .dat que se graficara

5.5.2. Menu

- Necesitamos un menu para pedir entradas al usuario, nuestro se vera de la siguiente manera:

Listing 17: Menu

```
    System.out.println("\n=======Menu Principal========");
    System.out.println("1. Ordenamiento por Insercin");
    System.out.println("2. Ordenamiento por Quicksort");
    System.out.println("3. Bsqueda Binaria Recursiva");
    System.out.println("4. Bsqueda Binaria Iterativa");
    System.out.println("5. Comparar Rendimientos");
    System.out.println("6. Salir");
    System.out.print("Seleccione una opcin: ");
```

- Adicionalmente, tendremos que mostrarle las opciones de ordenamiento para cada uno de los algoritmos:

Listing 18: Menu

```
        System.out.println("\n=======Ordenar por:========");
```

```
        System.out.println("1. cui");
        System.out.println("2. email");
        System.out.println("3. name");
        System.out.println("4. apellido paterno");
        System.out.println("5. apellido materno");
        System.out.println("6. F. nacimiento");
        System.out.println("7. Genero");
        System.out.println("8. Status");
```

- Con esto pediremos dos entradas al usuario, que seran almacenadas en dos variables de tipo entero

La estructura de ejecucion de nuestro menu, consta de switchs anidados, el primero incluye los elementos del primer menu, y el segundo incluye los elementos del segundo menu

5.5.3. Clase Test

- Ahora necesitamos una clase Test, que corra un algoritmo con casos de prueba
- Tenemos dos metodos principales, runAlgorithm y compareAlgorith
- La clase tiene los siguientes atributos:
  - String name, se refiere al algoritmo que se esta ejecutando
  - Reader.Student[] sujetosPrueba, es la copia de array para hacer los ordenamientos (Importante: Para evitar el conflicto de pasar objeto por referencia en lugar de valor, cada llamada a Test copiara un nuevo arreglo identico al original para asegurarnos de tener siempre un arreglo desordenado)

5.5.4. selectUser: Metodo auxiliar

- Para poder ejecutar el algoritmo correcto segun la entrada del usuario, necesitaremos un metodo que tome la entrada del usuario y corra el algoritmo correcto
- Las entradas para el metodo son, son dos enteros representando la eleccion 1 y eleccion 2 del usuario, y un arreglo de estudiantes para ordenar

Listing 19: Metodo auxiliar

```

static public void selectUser(int o1, int o2, Reader.Student[] muestra){
    int option = o1 * 10 + o2;
    switch (option) {
      case 11:
        QuickSort.cui(muestra, 0, muestra.length - 1);
        name = "CUI";
        break;
      case 12:
        QuickSort.email(muestra, 0, muestra.length - 1);
        name = "Email";
        break;
      case 13:
        QuickSort.name(muestra, 0, muestra.length - 1);
        name = "Nombre";
        break;
      case 14:
        QuickSort.lastNameF(muestra, 0, muestra.length - 1);
        name = "Apellido Paterno";
        break;
        ...
```

5.5.5. Metodo runAlgorithm

- El metodo se encarga de hacer un test a un determinado algoritmo

Listing 20: Testear algoritmo

```
static public void runAlgorithm(int o1, int o2){
  String data = "";
  for(int i = 0; i < 100; i++){
    if(i == 0)
      continue;
    Reader.Student[] muestra = Arrays.copyOf(sujetosPrueba, testCases[i]);
    long startTime = System.nanoTime();
    selectUser(o1, o2, muestra);
    long endTime = System.nanoTime();
    String time = Long.toString(endTime - startTime);
    data = data + "\n" + String.valueOf(testCases[i]) + "\t" + time;
  }
  try (BufferedWriter bw = new BufferedWriter(new FileWriter("./graphics/input/data.dat"))) {
    bw.write(data);
  } catch (IOException e) {
    e.printStackTrace();
  }
  String comando = "./graphics/graficar.sh \"" + name + "\" ./graphics/input/data.dat";
  try {
      ProcessBuilder builder = new ProcessBuilder();
      builder.command("sh", "-c", comando);
      Process proceso = builder.start();
  } catch (IOException e) {
      e.printStackTrace();
  }
}
```

- Como vemos, se recorre un ciclo for 100 veces, y en cada ciclo se llama a el metodo selectUser (que ejecutara el algoritmo correcto segun lo que el usuario ingrese) y guargara los datos de cada ejecucion en el String data
- Posteriormente se escribe un archivo llamado data.dat que contiene los datos de ejecucion del algoritmo
- Con esto, se procede a ejecutar por consola el comando que llamara a nuestro scritp anteriormente mostrado, para generar una grafica por pantalla

5.5.6. Metodo compareAlgorithm

- Para esto, necesitaremos hacer una doble implementacion del anterior metodo,
- Escribiremos dos archivos de dos algoritmos respectivos y luego ejecutaremos el script con esos dos parametros

Listing 21: Testear algoritmo

```
static public void compareAlgorithm(){
  String data1 = "";
  String data2 = "";
  for(int i = 0; i < 100; i++){
    if(i == 0)
      continue;
    Reader.Student[] muestra1 = Arrays.copyOf(sujetosPrueba, testCases[i]);
    Reader.Student[] muestra2 = Arrays.copyOf(muestra1, muestra1.length);
```

```
        long startTime1 = System.nanoTime();
        selectUser(1, 6, muestra1);
        long endTime1 = System.nanoTime();
        String time1 = Long.toString(endTime1 - startTime1);
        data1 = data1 + "\n" + String.valueOf(testCases[i]) + "\t" + time1;

        long startTime2 = System.nanoTime();
        selectUser(2, 6, muestra2);
        long endTime2 = System.nanoTime();
        String time2 = Long.toString(endTime2 - startTime2);
        data2 = data2 + "\n" + String.valueOf(testCases[i]) + "\t" + time2;
      }
      String file1 = "./graphics/input/InsertionSort.dat";
      try (BufferedWriter bw = new BufferedWriter(new FileWriter(file1))) {
        bw.write(data1);
      } catch (IOException e) {
        e.printStackTrace();
      }

      String file2 = "./graphics/input/QuickSort.dat";
      try (BufferedWriter bw = new BufferedWriter(new FileWriter(file2))) {
        bw.write(data2);
      } catch (IOException e) {
        e.printStackTrace();
      }

      String comando = "./graphics/graficar.sh \"Comparacion de algoritmos\" " + file1 + " " + file2;
      try {
          ProcessBuilder builder = new ProcessBuilder();
          builder.command("sh", "-c", comando);
          Process proceso = builder.start();
      } catch (IOException e) {
          e.printStackTrace();
      }
    }
```

- Luego de llamar a una comparacion, tendremos una grafica por pantalla que nos mostrara el comportamiento de los dos algoritmos de ordenamiento

]

5.5.7. Commits principales

Listing 22: Testear algoritmo

```
commit 36c13d21d090dfff3745e873cd488fb5f6bf99b5
Author: JhonatanDczel <jariasq@unsa.edu.pe>
Date:  Wed Oct 4 13:11:18 2023 -0500

    Comparation, corrigiendo errores
```

- Previamente se subio una version de la Clase Test, que ejecutaba los casos de prueba de un determinado algoritmo
- En esta version, se corrigieron errores de sintaxis y de ejecucion

Listing 23: Testear algoritmo

```
commit 5f67a4a5f222f5b6bf53675c01403ebfa3becc0a
```

```
Author: JhonatanDczel <jariasq@unsa.edu.pe>
Date:  Wed Oct 4 13:14:35 2023 -0500

    Implementacion de grafica automatizada
```

- En este commit se incluye la ejecucion del comando por consola para generar y abrir una gafica cada que se corra un algoritmo

Listing 24: Testear algoritmo

```
commit f679a725de0397dac72da1c384023d79fd4c6112
Author: ALVARO-QUISPE-UNSA <aquispecondo@unsa.edu.pe>
Date:  Wed Oct 4 12:39:44 2023 -0500

    Se  complet  el  men  de iterative binary search
```

- En este commit se da por completado el menu de ejecuciones

Listing 25: Testear algoritmo

```

commit f355c1bc818bca3a43a9fefa0deadc104b566e5e
Author: JhonatanDczel <jariasq@unsa.edu.pe>
Date:  Tue Oct 3 04:09:01 2023 -0500

    Graficadora: implementacion de grafica de comparacion para dos archivos
```

- Esta es la primera implementacion del script para graficar

Listing 26: Testear algoritmo

```

commit 7e25a93be444f6fc9ecfb8b6ad9a15d3e43aaa09 (HEAD -> main, origin/main, origin/HEAD)
Author: JhonatanDczel <jariasq@unsa.edu.pe>
Date:  Wed Oct 4 14:06:40 2023 -0500

    Graficadora completa
```

- Aqui se concluye con la parte grafica del proyecto, localizado en el archivo principal Compare.java

6. Ejecucion del programa

- A continuacion veremos la ejecucion del programa en la terminal:

6.1. Ejecución de Insertion

- A continuacion veremos la ejecucion del programa en la terminal:
- **Texto complementario**

Listing 27: Compilación y ejecución del código

```
 =======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 1

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
seleccione una opcin: 1
CUI: 10654321 Email: rmendozarodriguez@unsa.edu.pe Nombre: Raul A. Pat: Mendoza A. Mat: Rodriguez Fecha de Nacimiento: 1992-08-07 Genero: 0 Estado: 1
CUI: 11789012 Email: jperezgomez@unsa.edu.pe Nombre: Javier A. Pat: Perez A. Mat: Gomez Fecha de Nacimiento: 1990-12-30 Genero: 0 Estado: 1
CUI: 12876543 Email: cmendozarivera@unsa.edu.pe Nombre: Carlos A. Pat: Mendoza A. Mat: Rivera Fecha de Nacimiento: 1991-03-05 Genero: 0 Estado: 1
CUI: 13901234 Email: smartinezcruz@unsa.edu.pe Nombre: Sofia A. Pat: Martinez A. Mat: Cruz Fecha de Nacimiento: 1994-05-18 Genero: 0 Estado: 1
CUI: 14098765 Email: dlopezramos@unsa.edu.pe Nombre: David A. Pat: Lopez A. Mat: Ramos Fecha de Nacimiento: 1993-09-23 Genero: 0 Estado: 1
CUI: 15234567 Email: pmartinlopez@unsa.edu.pe Nombre: Paula A. Pat: Martin A. Mat: Lopez Fecha de Nacimiento: 1995-11-11 Genero: 0 Estado: 1
```

Listing 28: Compilación y ejecución del código

```
   javac Comparation.java
   java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 1

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
seleccione una opcin: 2
CUI: 23678901 Email: agomezlopez@unsa.edu.pe Nombre: Andrea A. Pat: Gomez A. Mat: Lopez Fecha de Nacimiento: 1999-04-05 Genero: 0 Estado: 1
CUI: 72456789 Email: amartinlopez@unsa.edu.pe Nombre: Alfredo A. Pat: Martin A. Mat: Lopez Fecha de Nacimiento: 1991-08-21 Genero: 0 Estado: 1
CUI: 78098765 Email: amendozamartinez@unsa.edu.pe Nombre: Alicia A. Pat: Mendoza A. Mat: Martinez Fecha de Nacimiento: 1991-08-07 Genero: 0 Estado: 1
CUI: 20987654 Email: amendozariver@unsa.edu.pe Nombre: Ana A. Pat: Mendoza A. Mat: Rivera Fecha de Nacimiento: 1997-03-25 Genero: 0 Estado: 1
```

```
CUI: 23345678 Email: cmartinezmartinez@unsa.edu.pe Nombre: Carlos A. Pat: Martinez A. Mat:
    Martinez Fecha de Nacimiento: 1996-11-25 Genero: 1 Estado: 1
CUI: 27012345 Email: cmendozagonzalez@unsa.edu.pe Nombre: Clara A. Pat: Mendoza A. Mat: Gonzalez
    Fecha de Nacimiento: 1986-12-21 Genero: 0 Estado: 0
```

Listing 29: Compilación y ejecución del código

```
   javac Comparation.java
   java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3. Bsqueda Binaria Recursiva
4. Bsqueda Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 1

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
seleccione una opcin: 3
CUI: 41098765 Email: amartinlopez@unsa.edu.pe Nombre: Adriana A. Pat: Martin A. Mat: Lopez Fecha
    de Nacimiento: 1993-03-30 Genero: 0 Estado: 1
CUI: 36567890 Email: amendozaramirez@unsa.edu.pe Nombre: Alberto A. Pat: Mendoza A. Mat: Ramirez
    Fecha de Nacimiento: 1991-05-14 Genero: 0 Estado: 1
CUI: 72456789 Email: amartinlopez@unsa.edu.pe Nombre: Alfredo A. Pat: Martin A. Mat: Lopez Fecha
    de Nacimiento: 1991-08-21 Genero: 0 Estado: 1
CUI: 78098765 Email: amendozamartinez@unsa.edu.pe Nombre: Alicia A. Pat: Mendoza A. Mat: Martinez
    Fecha de Nacimiento: 1991-08-07 Genero: 0 Estado: 1
CUI: 20987654 Email: amendozariver@unsa.edu.pe Nombre: Ana A. Pat: Mendoza A. Mat: Rivera Fecha
    de Nacimiento: 1997-03-25 Genero: 0 Estado: 1
```

Listing 30: Compilación y ejecución del código

```
   javac Comparation.java
   java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3. Bsqueda Binaria Recursiva
4. Bsqueda Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 1

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
seleccione una opcin: 4
CUI: 21789012 Email: rgarciafernandez@unsa.edu.pe Nombre: Ral A. Pat: Garcia A. Mat: Fernandez
```

```
    Fecha de Nacimiento: 1992-07-22 Genero: 1 Estado: 1
CUI: 25012345 Email: jgarciafernandez@unsa.edu.pe Nombre: Javier A. Pat: Garcia A. Mat: Fernandez
    Fecha de Nacimiento: 1998-02-17 Genero: 1 Estado: 1
CUI: 84901234 Email: jgarcialopez@unsa.edu.pe Nombre: Julia A. Pat: Garcia A. Mat: Lopez Fecha de
    Nacimiento: 1993-04-19 Genero: 0 Estado: 1
CUI: 23678901 Email: agomezlopez@unsa.edu.pe Nombre: Andrea A. Pat: Gomez A. Mat: Lopez Fecha de
    Nacimiento: 1999-04-05 Genero: 0 Estado: 1
CUI: 26678901 Email: tgomezperez@unsa.edu.pe Nombre: Toribio A. Pat: Gomez A. Mat: Perez Fecha de
    Nacimiento: 1995-10-15 Genero: 1 Estado: 1
CUI: 19876543 Email: jgonzalezvargas@unsa.edu.pe Nombre: Juan A. Pat: Gonzalez A. Mat: Vargas
    Fecha de Nacimiento: 1995-09-20 Genero: 1 Estado: 1
CUI: 20567890 Email: rgutierrezperez@unsa.edu.pe Nombre: Ricardo A. Pat: Gutierrez A. Mat: Perez
    Fecha de Nacimiento: 2001-11-15 Genero: 1 Estado: 1
```

Listing 31: Compilación y ejecución del código

```
   javac Comparation.java
   java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 1

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
seleccione una opcin: 5
CUI: 22345678 Email: rmartinezbustos@unsa.edu.pe Nombre: Raul A. Pat: Martinez A. Mat: Bustos
    Fecha de Nacimiento: 1988-10-30 Genero: 1 Estado: 1
CUI: 13901234 Email: smartinezcruz@unsa.edu.pe Nombre: Sofia A. Pat: Martinez A. Mat: Cruz Fecha
    de Nacimiento: 1994-05-18 Genero: 0 Estado: 1
CUI: 21789012 Email: rgarciafernandez@unsa.edu.pe Nombre: Ral A. Pat: Garcia A. Mat: Fernandez
    Fecha de Nacimiento: 1992-07-22 Genero: 1 Estado: 1
CUI: 25012345 Email: jgarciafernandez@unsa.edu.pe Nombre: Javier A. Pat: Garcia A. Mat: Fernandez
    Fecha de Nacimiento: 1998-02-17 Genero: 1 Estado: 1
CUI: 65678901 Email: rmartinezgarcia@unsa.edu.pe Nombre: Ral A. Pat: Martinez A. Mat: Garcia
    Fecha de Nacimiento: 1991-06-20 Genero: 0 Estado: 1
```

Listing 32: Compilación y ejecución del código

```
   javac Comparation.java
   java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 1

=======Ordenar por:========
1. cui
```

```
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
seleccione una opcin: 6
CUI: 27012345 Email: cmendozagonzalez@unsa.edu.pe Nombre: Clara A. Pat: Mendoza A. Mat: Gonzalez Fecha de Nacimiento: 1986-12-21 Genero: 0 Estado: 0
CUI: 24012345 Email: lmendozamendoza@unsa.edu.pe Nombre: Luis A. Pat: Mendoza A. Mat: Mendoza Fecha de Nacimiento: 1987-01-09 Genero: 1 Estado: 0
CUI: 27678901 Email: rmendozatorres@unsa.edu.pe Nombre: Ricardo A. Pat: Mendoza A. Mat: Torres Fecha de Nacimiento: 1988-08-03 Genero: 1 Estado: 1
CUI: 22345678 Email: rmartinezbustos@unsa.edu.pe Nombre: Raul A. Pat: Martinez A. Mat: Bustos Fecha de Nacimiento: 1988-10-30 Genero: 1 Estado: 1
CUI: 26012345 Email: amendozamartinez@unsa.edu.pe Nombre: Andrs A. Pat: Mendoza A. Mat: Martinez Fecha de Nacimiento: 1989-11-08 Genero: 1 Estado: 1
CUI: 22678901 Email: jlopezgomez@unsa.edu.pe Nombre: Julia A. Pat: Lopez A. Mat: Gomez Fecha de Nacimiento: 1990-03-07 Genero: 0 Estado: 0
```

Listing 33: Compilación y ejecución del código

```
    javac Comparation.java
    java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3. Bsqueda Binaria Recursiva
4. Bsqueda Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 1

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
seleccione una opcin: 7
CUI: 20123456 Email: cmartinezmora@unsa.edu.pe Nombre: Carla A. Pat: Martinez A. Mat: Mora Fecha de Nacimiento: 1998-04-12 Genero: 0 Estado: 1
CUI: 20987654 Email: amendozariver@unsa.edu.pe Nombre: Ana A. Pat: Mendoza A. Mat: Rivera Fecha de Nacimiento: 1997-03-25 Genero: 0 Estado: 1
CUI: 21456789 Email: mmartinezgomez@unsa.edu.pe Nombre: Mara A. Pat: Martinez A. Mat: Gomez Fecha de Nacimiento: 1994-12-03 Genero: 0 Estado: 1
CUI: 22012345 Email: lmendozatorres@unsa.edu.pe Nombre: Laura A. Pat: Mendoza A. Mat: Torres Fecha de Nacimiento: 1996-05-18 Genero: 0 Estado: 1
CUI: 22678901 Email: jlopezgomez@unsa.edu.pe Nombre: Julia A. Pat: Lopez A. Mat: Gomez Fecha de Nacimiento: 1990-03-07 Genero: 0 Estado: 0
UI: 26012345 Email: amendozamartinez@unsa.edu.pe Nombre: Andrs A. Pat: Mendoza A. Mat: Martinez Fecha de Nacimiento: 1989-11-08 Genero: 1 Estado: 1
CUI: 26678901 Email: tgomezperez@unsa.edu.pe Nombre: Toribio A. Pat: Gomez A. Mat: Perez Fecha de Nacimiento: 1995-10-15 Genero: 1 Estado: 1
CUI: 27678901 Email: rmendozatorres@unsa.edu.pe Nombre: Ricardo A. Pat: Mendoza A. Mat: Torres Fecha de Nacimiento: 1988-08-03 Genero: 1 Estado: 1
CUI: 28345678 Email: smartinezperez@unsa.edu.pe Nombre: Sergio A. Pat: Martinez A. Mat: Perez Fecha de Nacimiento: 1999-10-09 Genero: 1 Estado: 1
```

Listing 34: Compilación y ejecución del código

```
    javac Comparation.java
    java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 1

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
seleccione una opcin: 8
CUI: 20235691 Email: hlopeztorres@unsa.edu.pe Nombre: Hugo A. Pat: Lopez A. Mat: Torres Fecha de Nacimiento: 2020-06-07 Genero: 1 Estado: 0
CUI: 21345678 Email: jtorresvargas@unsa.edu.pe Nombre: Jorge A. Pat: Torres A. Mat: Vargas Fecha de Nacimiento: 2000-08-10 Genero: 1 Estado: 0
CUI: 22678901 Email: jlopezgomez@unsa.edu.pe Nombre: Julia A. Pat: Lopez A. Mat: Gomez Fecha de Nacimiento: 1990-03-07 Genero: 0 Estado: 0
CUI: 24012345 Email: lmendozamendoza@unsa.edu.pe Nombre: Luis A. Pat: Mendoza A. Mat: Mendoza Fecha de Nacimiento: 1987-01-09 Genero: 1 Estado: 0
CUI: 27012345 Email: cmendozagonzalez@unsa.edu.pe Nombre: Clara A. Pat: Mendoza A. Mat: Gonzalez Fecha de Nacimiento: 1986-12-21 Genero: 0 Estado: 0
CUI: 28678901 Email: jlopezlopez@unsa.edu.pe Nombre: Julio A. Pat: Lopez A. Mat: Lopez Fecha de Nacimiento: 1991-07-13 Genero: 1 Estado: 0
CUI: 19946837 Email: pmendozariver@unsa.edu.pe Nombre: Pedro A. Pat: Mendoza A. Mat: Rivera Fecha de Nacimiento: 1999-02-14 Genero: 1 Estado: 1
CUI: 19876543 Email: jgonzalezvargas@unsa.edu.pe Nombre: Juan A. Pat: Gonzalez A. Mat: Vargas Fecha de Nacimiento: 1995-09-20 Genero: 1 Estado: 1
```

6.2. Ejecución de QuickSort

- A continuacion veremos la ejecucion del programa en la terminal:
- **Accedemos por medio del menú**

Listing 35: Compilación y ejecución del código

```
    javac Comparation.java
    java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 2

=======Ordenar por:========
1. cui
2. email
3. name
```

```
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
1
CUI: 10654321 Email: rmendozarodriguez@unsa.edu.pe Nombre: Raul A. Pat: Mendoza A. Mat: Rodriguez Fecha de Nacimiento: 1992-08-07 Genero: 0 Estado: 1
CUI: 11789012 Email: jperezgomez@unsa.edu.pe Nombre: Javier A. Pat: Perez A. Mat: Gomez Fecha de Nacimiento: 1990-12-30 Genero: 0 Estado: 1
CUI: 12876543 Email: cmendozarivera@unsa.edu.pe Nombre: Carlos A. Pat: Mendoza A. Mat: Rivera Fecha de Nacimiento: 1991-03-05 Genero: 0 Estado: 1
CUI: 13901234 Email: smartinezcruz@unsa.edu.pe Nombre: Sofia A. Pat: Martinez A. Mat: Cruz Fecha de Nacimiento: 1994-05-18 Genero: 0 Estado: 1
```

Listing 36: Compilación y ejecución del código

```
    javac Comparation.java
    java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 2

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
2
CUI: 23678901 Email: agomezlopez@unsa.edu.pe Nombre: Andrea A. Pat: Gomez A. Mat: Lopez Fecha de Nacimiento: 1999-04-05 Genero: 0 Estado: 1
CUI: 95678901 Email: amartinezmendez@unsa.edu.pe Nombre: Andrea A. Pat: Martinez A. Mat: Mendez Fecha de Nacimiento: 1997-10-14 Genero: 0 Estado: 1
CUI: 23098765 Email: amartinezsanchez@unsa.edu.pe Nombre: Ana A. Pat: Martinez A. Mat: Sanchez Fecha de Nacimiento: 1996-12-13 Genero: 0 Estado: 1
CUI: 72456789 Email: amartinlopez@unsa.edu.pe Nombre: Alfredo A. Pat: Martin A. Mat: Lopez Fecha de Nacimiento: 1991-08-21 Genero: 0 Estado: 1
CUI: 41098765 Email: amartinlopez@unsa.edu.pe Nombre: Adriana A. Pat: Martin A. Mat: Lopez Fecha de Nacimiento: 1993-03-30 Genero: 0 Estado: 1
CUI: 78098765 Email: amendozamartinez@unsa.edu.pe Nombre: Alicia A. Pat: Mendoza A. Mat: Martinez Fecha de Nacimiento: 1991-08-07 Genero: 0 Estado: 1
CUI: 26012345 Email: amendozamartinez@unsa.edu.pe Nombre: Andrs A. Pat: Mendoza A. Mat: Martinez Fecha de Nacimiento: 1989-11-08 Genero: 1 Estado: 1
```

Listing 37: Compilación y ejecución del código

```
    javac Comparation.java
    java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
```

```
6. Salir
seleccione una opcin: 2

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
3
CUI: 41098765 Email: amartinlopez@unsa.edu.pe Nombre: Adriana A. Pat: Martin A. Mat: Lopez Fecha de Nacimiento: 1993-03-30 Genero: 0 Estado: 1
CUI: 36567890 Email: amendozaramirez@unsa.edu.pe Nombre: Alberto A. Pat: Mendoza A. Mat: Ramirez Fecha de Nacimiento: 1991-05-14 Genero: 0 Estado: 1
CUI: 72456789 Email: amartinlopez@unsa.edu.pe Nombre: Alfredo A. Pat: Martin A. Mat: Lopez Fecha de Nacimiento: 1991-08-21 Genero: 0 Estado: 1
CUI: 78098765 Email: amendozamartinez@unsa.edu.pe Nombre: Alicia A. Pat: Mendoza A. Mat: Martinez Fecha de Nacimiento: 1991-08-07 Genero: 0 Estado: 1
CUI: 20987654 Email: amendozariver@unsa.edu.pe Nombre: Ana A. Pat: Mendoza A. Mat: Rivera Fecha de Nacimiento: 1997-03-25 Genero: 0 Estado: 1
```

Listing 38: Compilación y ejecución del código

```
    javac Comparation.java
    java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 2

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
4
CUI: 25012345 Email: jgarciafernandez@unsa.edu.pe Nombre: Javier A. Pat: Garcia A. Mat: Fernandez Fecha de Nacimiento: 1998-02-17 Genero: 1 Estado: 1
CUI: 84901234 Email: jgarcialopez@unsa.edu.pe Nombre: Julia A. Pat: Garcia A. Mat: Lopez Fecha de Nacimiento: 1993-04-19 Genero: 0 Estado: 1
CUI: 21789012 Email: rgarciafernandez@unsa.edu.pe Nombre: Ral A. Pat: Garcia A. Mat: Fernandez Fecha de Nacimiento: 1992-07-22 Genero: 1 Estado: 1
CUI: 23678901 Email: agomezlopez@unsa.edu.pe Nombre: Andrea A. Pat: Gomez A. Mat: Lopez Fecha de Nacimiento: 1999-04-05 Genero: 0 Estado: 1
CUI: 26678901 Email: tgomezperez@unsa.edu.pe Nombre: Toribio A. Pat: Gomez A. Mat: Perez Fecha de Nacimiento: 1995-10-15 Genero: 1 Estado: 1
CUI: 19876543 Email: jgonzalezvargas@unsa.edu.pe Nombre: Juan A. Pat: Gonzalez A. Mat: Vargas Fecha de Nacimiento: 1995-09-20 Genero: 1 Estado: 1
CUI: 20567890 Email: rgutierrezperez@unsa.edu.pe Nombre: Ricardo A. Pat: Gutierrez A. Mat: Perez Fecha de Nacimiento: 2001-11-15 Genero: 1 Estado: 1
CUI: 28678901 Email: jlopezlopez@unsa.edu.pe Nombre: Julio A. Pat: Lopez A. Mat: Lopez Fecha de Nacimiento: 1991-07-13 Genero: 1 Estado: 0
```

Listing 39: Compilación y ejecución del código

```
   javac Comparation.java
   java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 2

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
5
CUI: 22345678 Email: rmartinezbustos@unsa.edu.pe Nombre: Raul A. Pat: Martinez A. Mat: Bustos Fecha de Nacimiento: 1988-10-30 Genero: 1 Estado: 1
CUI: 13901234 Email: smartinezcruz@unsa.edu.pe Nombre: Sofia A. Pat: Martinez A. Mat: Cruz Fecha de Nacimiento: 1994-05-18 Genero: 0 Estado: 1
CUI: 21789012 Email: rgarciafernandez@unsa.edu.pe Nombre: Ral A. Pat: Garcia A. Mat: Fernandez Fecha de Nacimiento: 1992-07-22 Genero: 1 Estado: 1
CUI: 25012345 Email: jgarciafernandez@unsa.edu.pe Nombre: Javier A. Pat: Garcia A. Mat: Fernandez Fecha de Nacimiento: 1998-02-17 Genero: 1 Estado: 1
CUI: 65678901 Email: rmartinezgarcia@unsa.edu.pe Nombre: Ral A. Pat: Martinez A. Mat: Garcia Fecha de Nacimiento: 1991-06-20 Genero: 0 Estado: 1
CUI: 18567890 Email: vmartinezgomez@unsa.edu.pe Nombre: Valeria A. Pat: Martinez A. Mat: Gomez Fecha de Nacimiento: 1997-07-03 Genero: 0 Estado: 1
```

Listing 40: Compilación y ejecución del código

```
   javac Comparation.java
   java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 2

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
6
CUI: 27012345 Email: cmendozagonzalez@unsa.edu.pe Nombre: Clara A. Pat: Mendoza A. Mat: Gonzalez Fecha de Nacimiento: 1986-12-21 Genero: 0 Estado: 0
CUI: 24012345 Email: lmendozamendoza@unsa.edu.pe Nombre: Luis A. Pat: Mendoza A. Mat: Mendoza Fecha de Nacimiento: 1987-01-09 Genero: 1 Estado: 0
CUI: 27678901 Email: rmendozatorres@unsa.edu.pe Nombre: Ricardo A. Pat: Mendoza A. Mat: Torres Fecha de Nacimiento: 1988-08-03 Genero: 1 Estado: 1
```

```
CUI: 22345678 Email: rmartinezbustos@unsa.edu.pe Nombre: Raul A. Pat: Martinez A. Mat: Bustos Fecha de Nacimiento: 1988-10-30 Genero: 1 Estado: 1
CUI: 26012345 Email: amendozamartinez@unsa.edu.pe Nombre: Andrs A. Pat: Mendoza A. Mat: Martinez Fecha de Nacimiento: 1989-11-08 Genero: 1 Estado: 1
CUI: 22678901 Email: jlopezgomez@unsa.edu.pe Nombre: Julia A. Pat: Lopez A. Mat: Gomez Fecha de Nacimiento: 1990-03-07 Genero: 0 Estado: 0
CUI: 52234567 Email: lmendozaperez@unsa.edu.pe Nombre: Luis A. Pat: Mendoza A. Mat: Perez Fecha de Nacimiento: 1990-04-03 Genero: 0 Estado: 1
CUI: 27345678 Email: vmartinezrodriguez@unsa.edu.pe Nombre: Veronica A. Pat: Martinez A. Mat: Rodriguez Fecha de Nacimiento: 1990-05-26 Genero: 0 Estado: 1
CUI: 26456789 Email: lmartinezperez@unsa.edu.pe Nombre: Laura A. Pat: Martinez A. Mat: Perez Fecha de Nacimiento: 1990-07-24 Genero: 0 Estado: 1
```

Listing 41: Compilación y ejecución del código

```
   javac Comparation.java
   java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
6. Salir
seleccione una opcin: 2

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
7
CUI: 78098765 Email: amendozamartinez@unsa.edu.pe Nombre: Alicia A. Pat: Mendoza A. Mat: Martinez Fecha de Nacimiento: 1991-08-07 Genero: 0 Estado: 1
CUI: 77901234 Email: cmartinezperez@unsa.edu.pe Nombre: Cristian A. Pat: Martinez A. Mat: Perez Fecha de Nacimiento: 1997-06-24 Genero: 0 Estado: 1
CUI: 20123456 Email: cmartinezmora@unsa.edu.pe Nombre: Carla A. Pat: Martinez A. Mat: Mora Fecha de Nacimiento: 1998-04-12 Genero: 0 Estado: 1
CUI: 76876543 Email: lmartinrodriguez@unsa.edu.pe Nombre: Lucio A. Pat: Martin A. Mat: Rodriguez Fecha de Nacimiento: 1994-04-11 Genero: 0 Estado: 1
CUI: 20987654 Email: amendozariver@unsa.edu.pe Nombre: Ana A. Pat: Mendoza A. Mat: Rivera Fecha de Nacimiento: 1997-03-25 Genero: 0 Estado: 1
CUI: 24012345 Email: lmendozamendoza@unsa.edu.pe Nombre: Luis A. Pat: Mendoza A. Mat: Mendoza Fecha de Nacimiento: 1987-01-09 Genero: 1 Estado: 0
CUI: 23345678 Email: cmartinezmartinez@unsa.edu.pe Nombre: Carlos A. Pat: Martinez A. Mat: Martinez Fecha de Nacimiento: 1996-11-25 Genero: 1 Estado: 1
CUI: 23012345 Email: pmendozarodriguez@unsa.edu.pe Nombre: Pablo A. Pat: Mendoza A. Mat: Rodriguez Fecha de Nacimiento: 1993-08-14 Genero: 1 Estado: 1
```

Listing 42: Compilación y ejecución del código

```
   javac Comparation.java
   java Comparation
=======Menu Principal========
1. Ordenamiento por Insercin
2. Ordenamiento por Quicksort
3.  Bsqueda  Binaria Recursiva
4.  Bsqueda  Binaria Iterativa
5. Comparar Rendimientos
```

```
6. Salir
seleccione una opcin: 2

=======Ordenar por:========
1. cui
2. email
3. name
4. apellido paterno
5. apellido materno
6. F. nacimiento
7. Genero
8. Status
8
CUI: 20235691 Email: hlopeztorres@unsa.edu.pe Nombre: Hugo A. Pat: Lopez A. Mat: Torres Fecha de Nacimiento: 2020-06-07 Genero: 1 Estado: 0
CUI: 27012345 Email: cmendozagonzalez@unsa.edu.pe Nombre: Clara A. Pat: Mendoza A. Mat: Gonzalez Fecha de Nacimiento: 1986-12-21 Genero: 0 Estado: 0
CUI: 24012345 Email: lmendozamendoza@unsa.edu.pe Nombre: Luis A. Pat: Mendoza A. Mat: Mendoza Fecha de Nacimiento: 1987-01-09 Genero: 1 Estado: 0
CUI: 22678901 Email: jlopezgomez@unsa.edu.pe Nombre: Julia A. Pat: Lopez A. Mat: Gomez Fecha de Nacimiento: 1990-03-07 Genero: 0 Estado: 0
CUI: 21345678 Email: jtorresvargas@unsa.edu.pe Nombre: Jorge A. Pat: Torres A. Mat: Vargas Fecha de Nacimiento: 2000-08-10 Genero: 1 Estado: 0
CUI: 28678901 Email: jlopezlopez@unsa.edu.pe Nombre: Julio A. Pat: Lopez A. Mat: Lopez Fecha de Nacimiento: 1991-07-13 Genero: 1 Estado: 0
CUI: 21456789 Email: mmartinezgomez@unsa.edu.pe Nombre: Mara A. Pat: Martinez A. Mat: Gomez Fecha de Nacimiento: 1994-12-03 Genero: 0 Estado: 1
CUI: 21789012 Email: rgarciafernandez@unsa.edu.pe Nombre: Ral A. Pat: Garcia A. Mat: Fernandez Fecha de Nacimiento: 1992-07-22 Genero: 1 Estado: 1
CUI: 22012345 Email: lmendozatorres@unsa.edu.pe Nombre: Laura A. Pat: Mendoza A. Mat: Torres Fecha de Nacimiento: 1996-05-18 Genero: 0 Estado: 1
```

7. Rúbricas

7.1. Rúbrica para el contenido del Informe y demostración

- El equipo debe marcar o dejar en blanco las celdas de la columna **Checklist** si se cumplio con el ítem correspondiente.
- El equipo debe autocalificarse en la columna **Equipo** de acuerdo a la siguiente tabla:

Tabla 1: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 2: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio de trabajo en el repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	3	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	3	
Total		20		18	