

Informe Práctica 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
Práctica 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 Práctica 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 línea por línea con **BufferedReader**
- Separar cada línea 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

```
1 private void readDataFile(){
2
3     BufferedReader reader = null;
4
5     String line = "";
6
7     String[] parts;
8
9     try {
10
11         FileReader fileReader = new FileReader("./reader/data.csv");
12
13         reader = new BufferedReader(fileReader);
14
15         while ((line = reader.readLine()) != null) {
16
17             parts = line.split(",");
18
19             //... creacin y seteo de Student
20
21         }
22     }catch (Exception e){
23
24         e.printStackTrace();
25
26     }finally{
27
28         //...cerrar reader
29
30     }
31 }
32
33 }
```

Listing 2: Conversión de ArrayList a arreglo

```
1 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.

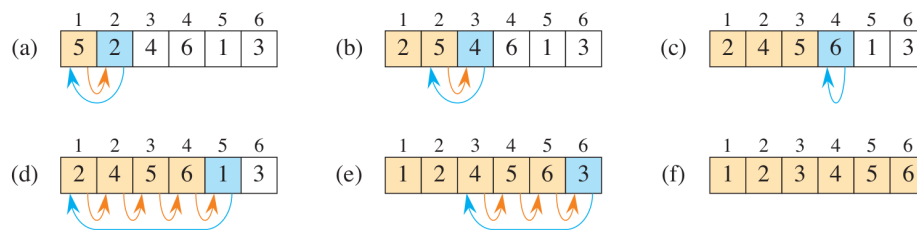


Figura 2: Iteraciones del ordenamiento por inserción

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 están 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 accesorios; 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 algoritmo de ordenamiento de Inserción donde nos aventuramos a diseñar el algoritmo usando nuestra propia lógica. Sin embargo, más adelante encontraremos formas más eficientes del algoritmo.

Listing 3: Commit: Se implementó Quicksort para cui

```
1 public static class InsertionAlgorithmCui (Student[] compaeritos ) {
2     int lowest;
3     Student pivot;
4     for(int i = 0; i < compaeritos.length; i++){
5         lowest = i;
6         pivot = compaeritos[i];
7         while((lowest > 0) && (compaeritos[lowest - 1].getCui() > pivot.getCui())){
8             compaeritos [lowest] = compaeritos[lowest - 1];
9             lowest--;
10        }
11        compaeritos [lowest] = pivot;
12    }
13 }
```

- 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

```
1 private static boolean compareString(String word1, String word2){
2     String lower1 = word1.toLowerCase();
3     String lower2 = word2.toLowerCase();
4     int length = lower1.length();
5     if(length > lower2.length()){
6         length = lower2.length();
7     }
8     int i = 0;
9     while(i < length){
10        if(lower1.charAt(i) == lower2.charAt(i)){
11            i++;
12        }else{
13            if(lower1.charAt(i) > lower2.charAt(i)){
14                return true;
15            }
16            return false;
17        }
18    }
19    return false;
20 }
21 }
```

- 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 conversion 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++){
```

```
6     j = i - 1;
7     pivot = classmates[i];
8     while(j > 0 && compareString(classmates[j].getEmail(), pivot.getEmail())){
9         classmates[j + 1] = classmates[j];
10        j--;
11    }
12    classmates[j + 1] = pivot;
13 }
14 }
```

- Hash:

-

Listing 8: TITULO000000

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 divide y vencerás. Consiste en seleccionar un elemento de la lista, llamado "pivote," 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

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

- 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 función «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();
```



```
21         s[i].setLastNameM(s[j].getLastNameM());
22         s[j].setLastNameM(aux);
23     }
24 }
25 aux = s[left].getLastNameM();
26 s[left].setLastNameM(s[j].getLastNameM());
27 s[j].setLastNameM(aux);
28 if (left < j - 1) lastNameM(s, left, j - 1);
29 if (j + 1 < right) lastNameM(s, j + 1, right);
30 }
```

- **Hash: 0c89058320e8f2dc1981a4666c3deb86df450702**
- Se solucionó 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

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

5.4. Búsqueda 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 creó «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

```
1 public static int cui (Reader.Student [] s, int x) {
2     int l, r;
3     l = 0;
4     r = s.length - 1;
5     while (l <= r) {
6         int m = l + ( r - l ) / 2;
7         if (s[m].getCui() == x) return m;
8         if (s[m].getCui() < x) l = m + 1;
9         else r = m - 1;
10    }
11    return -1;
12 }
```

- 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 búsqueda según email

```
1 public static int email (Reader.Student [] s, String x) {
2     int l = 0, r = s.length - 1;
3     while (l <= r) {
4         int m = l + (r - l) / 2;
5         if (s[m].getEmail().equals(x)) return m;
6         if (smallerThan(s[m].getEmail(), x)) l = m + 1;
7         else r = m - 1;
8     }
9     return -1;
10 }
11
12 public static boolean smallerThan(String w1, String w2) {
13     w1.toUpperCase();
14     w2.toUpperCase();
15     if (w1.compareTo(w2) <= 0) return true;
16     return false;
17 }
18
19
20 }
```

5.4.2. Forma recursiva

- Hash: **2318dc5647d46b889c72f61a09a9fd7b8f8f2f5a**
- Posteriormente se creó 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

```
1 package algorithms;
2
3 import reader.Reader;
4
5 public class RecursiveBinarySearch {
6     public static int cui (Reader.Student [] s, int l, int r) {
7         if (r >= l) {
8             int m = l + (r - l) / 2;
9             if (s[m].getCui() == x) return m;
10            if (s[m].getCui() > x) return cui(s, x, l, m - 1);
11            return cui(s, x, m + 1, r);
12        }
13        return -1;
14    }
15
16    public static int email (Reader.Student [] s, String x, int l, int r) {
17        if (r >= l) {
18            int m = l + (r - l) / 2;
19            if (s[m].getEmail().equals(x)) return m;
20            if ( smallerThan(s[m].getEmail(), x) ) return email(s, x, m + 1, r);
21            return email(s, x, l, m - 1);
22        }
23        return -1;
24    }
25
26    public static boolean smallerThan(String w1, String w2) {
27        w1.toUpperCase();
28        w2.toUpperCase();
29        if (w1.compareTo(w2) <= 0) return true;
30        return false;
31    }
32 }
```

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

```
1
2 #Definimos el nombre por defecto en caso de no haber ingresado un nombre en el segundo parametro
3 if [ "$outputFile" == ".png" ]; then
4     outputFile="Grafica.png"
5 fi
```

```

6
7 #Ahora definimos los nombres de los archivos
8 grafico1=$(echo "$2" | cut -c 18- | rev | cut -c 5- | rev)
9 grafico2=$(echo "$3" | cut -c 18- | rev | cut -c 5- | rev)
10 #Definimos la linea que se usara para la salida del plot
11 plotLine="plot \"$2\" w lp lw 3 lc rgb \"#62aef\" pt 6 title \"$grafico1\""
12 if [[ $3 != "" ]]; then
13     plotLine="$plotLine, \"$3\" w lp lw 3 lc rgb \"#ddf056\" pt 6 t \"$grafico2\""
14 fi
15
16 # aqui se ejecuta Gnuplot para generar el grfico y exportarlo
17 gnuplot <<-EOF
18 # Cambiamos el output de la terminal a png
19 set terminal pngcairo enhanced
20 set output "./graphics/imgs/$outputFile"
21
22 set title "$1" textcolor rgb "white" font "helvetica,12"
23 set key left box textcolor "white" font "helvetica,12"
24 set object 1 rectangle from screen 0,0 to screen 1,1 behind noclip fc rgb "#32363d" fillstyle solid
25     1.0
26 set border lc rgb "white" lw 2
27 set xlabel "Data count" font "helvetica,12" textcolor "white"
28 set ylabel "Time (ns)" font "helvetica,12" textcolor "white"
29 set tics font "helvetica,11" textcolor rgb "#faf95d"
30
31 # Se crea la grafica
32 $plotLine
33 EOF
34 xdg-open "./graphics/imgs/$outputFile"
35 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

```

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

```

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

Listing 18: Menu

```

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

```

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

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

La estructura de ejecución de nuestro menú, consta de switches anidados, el primero incluye los elementos del primer menú, y el segundo incluye los elementos del segundo menú

5.5.3. Clase Test

- Ahora necesitamos una clase Test, que corra un algoritmo con casos de prueba
- Tenemos dos métodos principales, `runAlgorithm` y `compareAlgorithm`
- La clase tiene los siguientes atributos:
 - String `name`, se refiere al algoritmo que se está 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 copiará un nuevo arreglo idéntico al original para asegurarnos de tener siempre un arreglo desordenado)

5.5.4. `selectUser`: Método auxiliar

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

Listing 19: Método auxiliar

```
1
2      static public void selectUser(int o1, int o2, Reader.Student[] muestra){
3          int option = o1 * 10 + o2;
4          switch (option) {
5              case 11:
6                  QuickSort.cui(muestra, 0, muestra.length - 1);
7                  name = "CUI";
8                  break;
9              case 12:
10                 QuickSort.email(muestra, 0, muestra.length - 1);
11                 name = "Email";
12                 break;
13             case 13:
14                 QuickSort.name(muestra, 0, muestra.length - 1);
15                 name = "Nombre";
16                 break;
17             case 14:
18                 QuickSort.lastNameF(muestra, 0, muestra.length - 1);
19                 name = "Apellido Paterno";
20                 break;
21             ...
```

5.5.5. Metodo runAlgorithm

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

Listing 20: Testear algoritmo

```
1 static public void runAlgorithm(int o1, int o2){
2     String data = "";
3     for(int i = 0; i < 100; i++){
4         if(i == 0)
5             continue;
6         Reader.Student[] muestra = Arrays.copyOf(sujetosPrueba, testCases[i]);
7         long startTime = System.nanoTime();
8         selectUser(o1, o2, muestra);
9         long endTime = System.nanoTime();
10        String time = Long.toString(endTime - startTime);
11        data = data + "\n" + String.valueOf(testCases[i]) + "\t" + time;
12    }
13    try (BufferedWriter bw = new BufferedWriter(new FileWriter("./graphics/input/data.dat"))) {
14        bw.write(data);
15    } catch (IOException e) {
16        e.printStackTrace();
17    }
18    String comando = "./graphics/graficar.sh \"" + name + "\" ./graphics/input/data.dat";
19    try {
20        ProcessBuilder builder = new ProcessBuilder();
21        builder.command("sh", "-c", comando);
22        Process proceso = builder.start();
23    } catch (IOException e) {
24        e.printStackTrace();
25    }
26 }
```

- 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 scrip 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

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

```

9      long startTime1 = System.nanoTime();
10     selectUser(1, 6, muestra1);
11     long endTime1 = System.nanoTime();
12     String time1 = Long.toString(endTime1 - startTime1);
13     data1 = data1 + "\n" + String.valueOf(testCases[i]) + "\t" + time1;
14
15     long startTime2 = System.nanoTime();
16     selectUser(2, 6, muestra2);
17     long endTime2 = System.nanoTime();
18     String time2 = Long.toString(endTime2 - startTime2);
19     data2 = data2 + "\n" + String.valueOf(testCases[i]) + "\t" + time2;
20 }
21 String file1 = "./graphics/input/InsertionSort.dat";
22 try (BufferedWriter bw = new BufferedWriter(new FileWriter(file1))) {
23     bw.write(data1);
24 } catch (IOException e) {
25     e.printStackTrace();
26 }
27
28 String file2 = "./graphics/input/QuickSort.dat";
29 try (BufferedWriter bw = new BufferedWriter(new FileWriter(file2))) {
30     bw.write(data2);
31 } catch (IOException e) {
32     e.printStackTrace();
33 }
34
35 String comando = "./graphics/graficar.sh \"Comparacion de algoritmos\" " + file1 + " " + file2;
36 try {
37     ProcessBuilder builder = new ProcessBuilder();
38     builder.command("sh", "-c", comando);
39     Process proceso = builder.start();
40 } catch (IOException e) {
41     e.printStackTrace();
42 }
43 }

```

- 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

```

1 commit 36c13d21d090dfff3745e873cd488fb5f6bf99b5
2 Author: JhonatanDczel <jariasq@unsa.edu.pe>
3 Date: Wed Oct 4 13:11:18 2023 -0500
4
5     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

```

1 commit 5f67a4a5f222f5b6bf53675c01403ebfa3becc0a

```

```
2 Author: JhonatanDczel <jariasq@unsa.edu.pe>
3 Date: Wed Oct 4 13:14:35 2023 -0500
4
5 Implementacion de grafica automatizada
```

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

Listing 24: Testear algoritmo

```
1 commit f679a725de0397dac72da1c384023d79fd4c6112
2 Author: ALVARO-QUISPE-UNSA <aquispecondo@unsa.edu.pe>
3 Date: Wed Oct 4 12:39:44 2023 -0500
4
5 Se complet el men de iterative binary search
```

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

Listing 25: Testear algoritmo

```
1
2 commit f355c1bc818bca3a43a9fefa0deadc104b566e5e
3 Author: JhonatanDczel <jariasq@unsa.edu.pe>
4 Date: Tue Oct 3 04:09:01 2023 -0500
5
6 Graficadora: implementacion de grafica de comparacion para dos archivos
```

- Esta es la primera implementacion del script para graficar

Listing 26: Testear algoritmo

```
1
2 commit 7e25a93be444f6fc9ecfb8b6ad9a15d3e43aaa09 (HEAD -> main, origin/main, origin/HEAD)
3 Author: JhonatanDczel <jariasq@unsa.edu.pe>
4 Date: Wed Oct 4 14:06:40 2023 -0500
5
6 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

```
7  =====Menu Principal=====
8  1. Ordenamiento por Insercin
9  2. Ordenamiento por Quicksort
10 3. Bsqueda Binaria Recursiva
11 4. Bsqueda Binaria Iterativa
12 5. Comparar Rendimientos
13 6. Salir
14 seleccione una opcin: 1
15
16 =====Ordenar por:=====
17 1. cui
18 2. email
19 3. name
20 4. apellido paterno
21 5. apellido materno
22 6. F. nacimiento
23 7. Genero
24 8. Status
25 seleccione una opcin: 1
26 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
27 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
28 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
29 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
30 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
31 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

```
33 javac Comparation.java
34 java Comparation
35 =====Menu Principal=====
36 1. Ordenamiento por Insercin
37 2. Ordenamiento por Quicksort
38 3. Bsqueda Binaria Recursiva
39 4. Bsqueda Binaria Iterativa
40 5. Comparar Rendimientos
41 6. Salir
42 seleccione una opcin: 1
43
44 =====Ordenar por:=====
45 1. cui
46 2. email
47 3. name
48 4. apellido paterno
49 5. apellido materno
50 6. F. nacimiento
51 7. Genero
52 8. Status
53 seleccione una opcin: 2
54 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
55 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
56 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
57 CUI: 20987654 Email: amendozarivera@unsa.edu.pe Nombre: Ana A. Pat: Mendoza A. Mat: Rivera Fecha
    de Nacimiento: 1997-03-25 Genero: 0 Estado: 1
```

```
58 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
59 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

```
61     javac Comparation.java
62     java Comparation
63     =====Menu Principal=====
64     1. Ordenamiento por Insercin
65     2. Ordenamiento por Quicksort
66     3. Bsqueda Binaria Recursiva
67     4. Bsqueda Binaria Iterativa
68     5. Comparar Rendimientos
69     6. Salir
70     seleccione una opcin: 1
71
72     =====Ordenar por:=====
73     1. cui
74     2. email
75     3. name
76     4. apellido paterno
77     5. apellido materno
78     6. F. nacimiento
79     7. Genero
80     8. Status
81     seleccione una opcin: 3
82     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
83     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
84     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
85     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
86     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

```
88     javac Comparation.java
89     java Comparation
90     =====Menu Principal=====
91     1. Ordenamiento por Insercin
92     2. Ordenamiento por Quicksort
93     3. Bsqueda Binaria Recursiva
94     4. Bsqueda Binaria Iterativa
95     5. Comparar Rendimientos
96     6. Salir
97     seleccione una opcin: 1
98
99     =====Ordenar por:=====
100    1. cui
101    2. email
102    3. name
103    4. apellido paterno
104    5. apellido materno
105    6. F. nacimiento
106    7. Genero
107    8. Status
108    seleccione una opcin: 4
109    CUI: 21789012 Email: rgarciafernandez@unsa.edu.pe Nombre: Ral A. Pat: Garcia A. Mat: Fernandez
```

```
110 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
111 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
112 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
113 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
114 Nacimiento: 1995-10-15 Genero: 1 Estado: 1
    CUI: 19876543 Email: jgonzalezvargas@unsa.edu.pe Nombre: Juan A. Pat: Gonzalez A. Mat: Vargas
115 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

```
117 javac Comparation.java
118 java Comparation
119 =====Menu Principal=====
120 1. Ordenamiento por Insercin
121 2. Ordenamiento por Quicksort
122 3. Bsqueda Binaria Recursiva
123 4. Bsqueda Binaria Iterativa
124 5. Comparar Rendimientos
125 6. Salir
126 seleccione una opcin: 1
127
128 =====Ordenar por:=====
129 1. cui
130 2. email
131 3. name
132 4. apellido paterno
133 5. apellido materno
134 6. F. nacimiento
135 7. Genero
136 8. Status
137 seleccione una opcin: 5
138 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
139 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
140 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
141 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
142 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

```
144 javac Comparation.java
145 java Comparation
146 =====Menu Principal=====
147 1. Ordenamiento por Insercin
148 2. Ordenamiento por Quicksort
149 3. Bsqueda Binaria Recursiva
150 4. Bsqueda Binaria Iterativa
151 5. Comparar Rendimientos
152 6. Salir
153 seleccione una opcin: 1
154
155 =====Ordenar por:=====
156 1. cui
```

```
157 2. email
158 3. name
159 4. apellido paterno
160 5. apellido materno
161 6. F. nacimiento
162 7. Genero
163 8. Status
164 seleccione una opcin: 6
165 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
166 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
167 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
168 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
169 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
170 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

```
172     javac Comparation.java
173     java Comparation
174     =====Menu Principal=====
175     1. Ordenamiento por Insercin
176     2. Ordenamiento por Quicksort
177     3. Bsqueda Binaria Recursiva
178     4. Bsqueda Binaria Iterativa
179     5. Comparar Rendimientos
180     6. Salir
181     seleccione una opcin: 1
182
183     =====Ordenar por:=====
184     1. cui
185     2. email
186     3. name
187     4. apellido paterno
188     5. apellido materno
189     6. F. nacimiento
190     7. Genero
191     8. Status
192     seleccione una opcin: 7
193     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
194     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
195     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
196     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
197     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
198     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
199     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
200     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
201     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

```
203     javac Comparation.java
204     java Comparation
205     =====Menu Principal=====
206     1. Ordenamiento por Insercin
207     2. Ordenamiento por Quicksort
208     3. Bsqueda Binaria Recursiva
209     4. Bsqueda Binaria Iterativa
210     5. Comparar Rendimientos
211     6. Salir
212     seleccione una opcin: 1
213
214     =====Ordenar por:=====
215     1. cui
216     2. email
217     3. name
218     4. apellido paterno
219     5. apellido materno
220     6. F. nacimiento
221     7. Genero
222     8. Status
223     seleccione una opcin: 8
224     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
225     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
226     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
227     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
228     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
229     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
230     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
231     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

```
233     javac Comparation.java
234     java Comparation
235     =====Menu Principal=====
236     1. Ordenamiento por Insercin
237     2. Ordenamiento por Quicksort
238     3. Bsqueda Binaria Recursiva
239     4. Bsqueda Binaria Iterativa
240     5. Comparar Rendimientos
241     6. Salir
242     seleccione una opcin: 2
243
244     =====Ordenar por:=====
245     1. cui
246     2. email
247     3. name
```

```
248 4. apellido paterno
249 5. apellido materno
250 6. F. nacimiento
251 7. Genero
252 8. Status
253 1
254 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
255 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
256 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
257 CUI: 13901234 Email: smartinezacruz@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

```
259     javac Comparation.java
260     java Comparation
261     =====Menu Principal=====
262     1. Ordenamiento por Insercin
263     2. Ordenamiento por Quicksort
264     3. Bsqueda Binaria Recursiva
265     4. Bsqueda Binaria Iterativa
266     5. Comparar Rendimientos
267     6. Salir
268     seleccione una opcin: 2
269
270     =====Ordenar por:=====
271     1. cui
272     2. email
273     3. name
274     4. apellido paterno
275     5. apellido materno
276     6. F. nacimiento
277     7. Genero
278     8. Status
279     2
280 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
281 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
282 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
283 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
284 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
285 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
286 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

```
288     javac Comparation.java
289     java Comparation
290     =====Menu Principal=====
291     1. Ordenamiento por Insercin
292     2. Ordenamiento por Quicksort
293     3. Bsqueda Binaria Recursiva
294     4. Bsqueda Binaria Iterativa
295     5. Comparar Rendimientos
```

```
296 6. Salir
297 seleccione una opcin: 2
298
299 =====Ordenar por:=====
300 1. cui
301 2. email
302 3. name
303 4. apellido paterno
304 5. apellido materno
305 6. F. nacimiento
306 7. Genero
307 8. Status
308 3
309 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
310 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
311 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
312 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
313 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

```
315 javac Comparation.java
316 java Comparation
317 =====Menu Principal=====
318 1. Ordenamiento por Insercin
319 2. Ordenamiento por Quicksort
320 3. Bsqueda Binaria Recursiva
321 4. Bsqueda Binaria Iterativa
322 5. Comparar Rendimientos
323 6. Salir
324 seleccione una opcin: 2
325
326 =====Ordenar por:=====
327 1. cui
328 2. email
329 3. name
330 4. apellido paterno
331 5. apellido materno
332 6. F. nacimiento
333 7. Genero
334 8. Status
335 4
336 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
337 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
338 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
339 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
340 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
341 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
342 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
343 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

```
345     javac Comparation.java
346     java Comparation
347     =====Menu Principal=====
348     1. Ordenamiento por Insercin
349     2. Ordenamiento por Quicksort
350     3. Bsqueda Binaria Recursiva
351     4. Bsqueda Binaria Iterativa
352     5. Comparar Rendimientos
353     6. Salir
354     seleccione una opcin: 2
355
356     =====Ordenar por:=====
357     1. cui
358     2. email
359     3. name
360     4. apellido paterno
361     5. apellido materno
362     6. F. nacimiento
363     7. Genero
364     8. Status
365     5
366     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
367     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
368     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
369     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
370     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
371     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

```
373     javac Comparation.java
374     java Comparation
375     =====Menu Principal=====
376     1. Ordenamiento por Insercin
377     2. Ordenamiento por Quicksort
378     3. Bsqueda Binaria Recursiva
379     4. Bsqueda Binaria Iterativa
380     5. Comparar Rendimientos
381     6. Salir
382     seleccione una opcin: 2
383
384     =====Ordenar por:=====
385     1. cui
386     2. email
387     3. name
388     4. apellido paterno
389     5. apellido materno
390     6. F. nacimiento
391     7. Genero
392     8. Status
393     6
394     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
395     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
396     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
```



```
397 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
398 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
399 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
400 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
401 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
402 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

```
404     javac Comparation.java
405     java Comparation
406     =====Menu Principal=====
407     1. Ordenamiento por Insercin
408     2. Ordenamiento por Quicksort
409     3. Bsqueda Binaria Recursiva
410     4. Bsqueda Binaria Iterativa
411     5. Comparar Rendimientos
412     6. Salir
413     seleccione una opcin: 2
414
415     =====Ordenar por:=====
416     1. cui
417     2. email
418     3. name
419     4. apellido paterno
420     5. apellido materno
421     6. F. nacimiento
422     7. Genero
423     8. Status
424     7
425 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
426 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
427 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
428 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
429 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
430 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
431 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
432 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

```
434     javac Comparation.java
435     java Comparation
436     =====Menu Principal=====
437     1. Ordenamiento por Insercin
438     2. Ordenamiento por Quicksort
439     3. Bsqueda Binaria Recursiva
440     4. Bsqueda Binaria Iterativa
441     5. Comparar Rendimientos
```

```
442 6. Salir
443 seleccione una opcin: 2
444
445 =====Ordenar por:=====
446 1. cui
447 2. email
448 3. name
449 4. apellido paterno
450 5. apellido materno
451 6. F. nacimiento
452 7. Genero
453 8. Status
454 8
455 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
456 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
457 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
458 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
459 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
460 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
461 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
462 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
463 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
```

6.3. Ejecución de Insertion

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

Listing 43: Compilación y ejecución del código

```
465 ejecucin
```

6.4. Ejecución de Insertion

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

Listing 44: Compilación y ejecución del código

```
466 ejecucin
```

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 cumple 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 están 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	