Informe de Laboratorio 01

Tema: Git y GitHub

Nota

Estudiante	Escuela	Asignatura
Juan Perez Luna jperez@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación Semestre: I Código: 20231001

Laboratorio	Tema	Duración
01	Git y GitHub	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - A	Del 10 Abril 2023	Al 17 Abril 2023

1. Tarea

- Implementar el algoritmo de ordenamiento por inserción en Java para mostrar el comportamiento de su complejidad para el peor caso.
- Utilizar Git para evidenciar su trabajo.
- Enviar trabajo al profesor en un repositorio GitHub Privado, dándole permisos como colaborador.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Ubuntu GNU Linux 23 lunar 64 bits Kernell 6.2.
- VIM 9.0.
- OpenJDK 64-Bits 17.0.7.
- Git 2.39.2.
- Cuenta en GitHub con el correo institucional.
- Programación Orientada a Objetos.
- Algoritmo de ordenamiento por inserción

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/rescobedoq/programacion.git`
- URL para el laboratorio 01 en el Repositorio GitHub.
- `https://github.com/rescobedoq/programacion/tree/main/lab01`

4. Actividades con el repositorio GitHub

4.1. Creando e inicializando repositorio GitHub

- Como es el primer laboratorio se creo el repositorio GitHub.
- Se realizaron los siguientes comandos en la computadora:

Listing 1: Creando directorio de trabajo

```
$ mkdir -p $HOME/rescobedoq/
```

Listing 2: Dirijiéndonos al directorio de trabajo

```
$ cd $HOME/rescobedoq/
```

Listing 3: Creando directorio para repositorio GitHub

```
$ mkdir -p $HOME/rescobedoq/programacion
```

Listing 4: Inicializando directorio para repositorio GitHub

```
$ cd $HOME/rescobedoq/programacion
$ echo "# programacion" >> README.md
$ git init
$ git config --global user.name "Richart Smith Escobedo Quispe"
$ git config --global user.email rescobedoq@gmail.com
$ git add README.md
$ git commit -m "first commit"
$ git branch -M main
$ git remote add origin https://github.com/rescobedoq/programacion.git
$ git push -u origin main
```

4.2. Commits

Listing 5: Primer Commit Creando carpeta/archivo para laboratorio 01

```
$ mkdir lab01
$ touch lab01/Insertion.java
$ git add .
$ git commit -m "Creando carpeta/archivo para laboratorio 01"
$ git push -u origin main
```

- Se creo el archivo **.gitignore** para no considerar los archivos ***.class** que son innecesarios hacer seguimiento.

Listing 6: Creando .gitignore

```
$ vim lab01/.gitignore
```

Listing 7: lab01/.gitignore

```
*.class
```

Listing 8: Commit: Creando .gitignore para archivos *.class

```
$ git add .
$ git commit -m "Creando .gitignore para archivos *.class"
$ git push -u origin main
```

- Para el siguiente commit se implemento el algoritmo de ordenamiento por Inserción, se imprime el arreglo caso definido en el mismo código.
- El pseudocódigo utilizado es el siguiente:

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

Listing 9: Creando .gitignore

```
$ vim lab01/Insertion.java
```

Listing 10: Insertion.java

```
public class Insertion {

   public static void main(String[] args) {
      int[] lista = {5, 2, 4, 6, 1, 3};
      imprimirArreglo(lista);
      insertionSort(lista);
   }

   public static void insertionSort(int[] A) {
      int key;
      int i;
      for (int j=1; j<A.length; j=j+1) {
         key = A[j];
         //Insertar A[j] en la secuencia ordenada A[1..j-1]
         i=j-1;
         while(i>-1 && A[i]>key) {
            A[i+1] = A[i];
            i = i-1;
         }
         imprimirArreglo(A);
         A[i+1] = key;
      }
   }

   public static void imprimirArreglo(int[] lista){
      System.out.println("");
      for (int x=0; x<lista.length; x++) {
         System.out.print(lista[x] + " ");
      }
   }
}
```

Listing 11: Compilando y probando código

```
$ cd lab01
$ javac Insertion.java
$ java Insertion
5 2 4 6 1 3
5 5 4 6 1 3
2 5 5 6 1 3
2 4 5 6 1 3
2 2 4 5 6 3
1 2 4 4 5 6
```

Listing 12: Commit: Probando algoritmo de Inserción con arreglo

```
$ git add .
$ git commit -m "Probando algoritmo de Insercion con arreglo"
$ git push -u origin main
```

4.3. Estructura de laboratorio 01

- El contenido que se entrega en este laboratorio es el siguiente:

```
.
|--- bin
|   |--- activate
|   |--- activate.csh
|   |--- activate.fish
|   |--- activate.nu
|   |--- activate.ps1
|   |--- activate_this.py
|   |--- deactivate.nu
|   |--- pip
|   |--- pip-3.9
|   |--- pip3
|   |--- pip3.9
|   |--- python -> /usr/local/opt/python@3.9/bin/python3.9
|   |--- python3 -> python
|   |--- python3.9 -> python
|   |--- wheel
|   |--- wheel-3.9
|   |--- wheel3
|   |--- wheel3.9
|--- lib
|   |--- python3.9
|--- pyvenv.cfg
|--- text.txt

3 directories, 20 files
```

5. Rúbricas

5.1. Entregable Informe

Tabla 1: Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

5.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplio con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos lo items.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 2: 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 3: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su 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			
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		
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			
Total		20		12	

6. Referencias

- https://www.w3schools.com/java/default.asp
- https://www.geeksforgeeks.org/insertion-sort/