

Informe de Laboratorio 01

Tema: Git, HTML y CSS

Nota

Estudiante	Escuela	Asignatura
Paulo Andre hidalgo Chinchay phidalgo@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación web 2 Semestre: III Código: 20223011

Laboratorio	Tema	Duración
01	Git, HTML y CSS	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - A	Del 2 Junio 2023	Al 5 Junio 2023

1. Tarea

- Crear una pagina propia con HTML y CSS que incluya nuestra presentación personal al igual que imágenes y links.
- Utilizar Git para evidenciar subir el trabajo.
- Crear un repositorio privado.
- Hacer que el docente sea colaborador de este repositorio para que pueda ser revisado.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Ubuntu GNU Linux 23 lunar 64 bits Kernell 6.2.
- Sistema Operativo Windows 11 pro versión 22H2 de 64 bits.
- VIM 9.0.
- Git 2.39.2.
- Visual Studio Code 1.78.2
- Cuenta en GitHub con el correo institucional.
- Tutoriales de w3c de HTML y CSS
- <https://www.w3.org/Style/Examples/011/firstcss.es.html>

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- <https://github.com/PauloUNSA/pw2-lab-c-23a.git>
- URL para el laboratorio 01 en el Repositorio GitHub.
- <https://github.com/PauloUNSA/pw2-lab-c-23a/tree/main/lab1>

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: Dirigiéndonos 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

```
1 public class Insertion {
2
3     public static void main(String[] args) {
4         int[] lista = {5, 2, 4, 6, 1, 3};
5         imprimirArreglo(lista);
6         insertionSort(lista);
7     }
8
9     public static void insertionSort(int[] A) {
10        int key;
11        int i;
12        for (int j=1; j<A.length; j=j+1) {
13            key = A[j];
14            //Insertar A[j] en la secuencia ordenada A[1..j-1]
15            i=j-1;
16            while(i>-1 && A[i]>key) {
17                A[i+1] = A[i];
18                i = i-1;
19            }
20            imprimirArreglo(A);
21            A[i+1] = key;
22        }
23    }
24
25    public static void imprimirArreglo(int[] lista){
26        System.out.println("");
27        for (int x=0; x<lista.length; x++) {
28            System.out.print(lista[x] + " ");
29        }
30    }
31 }
```

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:

```
lab01/  
|--- Insertion.java  
|--- latex  
|   |--- img  
|   |   |--- logo_abet.png  
|   |   |--- logo_episunsa.png  
|   |   |--- logo_unsa.jpg  
|   |   |--- pseudocodigo_insercion.png  
|--- programacion_lab01_rescobedoq_v1.0.pdf  
|--- programacion_lab01_rescobedoq_v1.0.tex  
|--- src  
|   |--- Insertion01.java
```

5. Pregunta: ¿Cuál es el comportamiento del algoritmo de ordenamiento por inserción?

- El algoritmo muestra un comportamiento cuadrático de $O(n^2)$.
- Se trabajaron los peores casos desde un arreglo de tamaño 1 hasta N.
- Para obtener un gráfico ideal se utilizó $N=10,000$.

6. Rúbricas

6.1. Entregable Informe

Tabla 1: Tipo de Informe

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

6.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumple 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 los ítems.
- 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 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			
Total		20		12	

7. Referencias

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