Informe de Laboratorio 01

Tema: Git y GitHub

Nota

Estudiante	Escuela	Asignatura
Jose Carlos Chino Nina jchino@ulasalle.edu.pe	Carrera Profesional de Ingeniería de Software	Lenguaje de Programación Semestre: I Código: 20231001

Laboratorio	Tema	Duración
01	Git y GitHub	06 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2025 - B	Del 21 Agosto 2025	Al 26 Agosto 2025

1. Tarea

- Explore un problema intermedio en el sitio web OmegaUp.
- Envíe al tablón del classroom que usted estudiará ese ejercicio.
- Cree su repositorio remoto privado para el curso como se explicó en clase. Ejemplo: https://github.com/.../lp2.git
- Cree su repositorio local Git.
- Cree el archivo README.md para el repositorio.
- Cree el archivo lab01/README.md para la descripción del laboratorio 1.
- Cree el archivo .gitignore para evitar envío de archivos innecesarios.
- Cree la solución a su problema seleccionado en el archivo lab01/problema.cpp.
- Compile y ejecute su programa.
- Sincronice su repositorio remoto desde su repositorio git local.
- Entregable: URL de su repositorio donde se encuentre el laboratorio 1, que resuelve la tarea, de acuerdo con las especificaciones dadas por el docente.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Windows 11 Version 23H2
- OpenJDK 64-Bits 17.0.7.
- Git 2.39.2.
- Cuenta en GitHub con el correo institucional.
- Visual Studio Code.
- Compilador GCC/G++.
- GitHub.

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/Jchino24/lp2.git`
- URL para el laboratorio 01 en el Repositorio GitHub.
- `https://github.com/Jchino24/lp2/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/Jchino24/
```

Listing 2: Dirijíéndonos al directorio de trabajo

```
$ cd $HOME/Jchino24/
```

Listing 3: Creando directorio para repositorio GitHub

```
$ mkdir -p $HOME/Jchino24/lp
```

Listing 4: Inicializando directorio para repositorio GitHub

```
$ cd $HOME/Jchino24/lp
$ echo "# programacion" >> README.md
$ git init
$ git config --global user.name "Jose Carlos Chino Nina"
$ git config --global user.email jchino@ulasalle.edu.pe
$ git add README.md
$ git commit -m "first commit"
```

```
$ git branch -M main
$ git remote add origin https://github.com/Jchino24/lp.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:

```
1   node_modules/
2   *.env
3   *.log
4   *.out
5   *.exe
6   *.bat
```

Listing 9: Problema.cpp

```cpp
#include <iostream>

using namespace std;

int main(){

    int cantidad;
    cout << "Ingrese la cantidad de monedas: ";
    cin >> cantidad;

    if (cantidad < 0) {
        cout << "La cantidad no puede ser negativa." << endl;
        return 1;
    }

    int monedas_500 = cantidad / 500;
    cantidad %= 500;

    int monedas_200 = cantidad / 200;
    cantidad %= 200;

    int monedas_100 = cantidad / 100;
    cantidad %= 100;

    int monedas_50 = cantidad / 50;
    cantidad %= 50;

    int monedas_25 = cantidad / 25;
    cantidad %= 25;

    int monedas_10 = cantidad / 10;
    cantidad %= 10;

    int monedas_5 = cantidad / 5;
    cantidad %= 5;

    int monedas_1 = cantidad;

    cout << "De 500 hay " << monedas_500 << endl;
    cout << "De 200 hay " << monedas_200 << endl;
    cout << "De 100 hay " << monedas_100 << endl;
    cout << "De 50 hay " << monedas_50 << endl;
    cout << "De 25 hay " << monedas_25 << endl;
    cout << "De 10 hay " << monedas_10 << endl;
    cout << "De 5 hay " << monedas_5 << endl;
    cout << "De 1 hay " << monedas_1 << endl;

    return 0;

}
```

Listing 10: Compilando y probando código

```
g++ problema.cpp -o problema.exe
  .\problema.exe
Ingrese la cantidad de monedas: 2562
  De 500 hay 5
  De 200 hay 0
  De 100 hay 0
  De 50 hay 1
  De 25 hay 0
  De 10 hay 1
  De 5 hay 0
  De 1 hay 2
```

4.3. Estructura de laboratorio 01

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

```
.
|--- lab01 [DIRECTORY]
     |--- Ejemplo.png
     |--- README.md
     |--- problema.cpp
|--- .gitignore
|--- README.md
|--- me.cpp
|--- latex [DIRECTORY]
     |--- jperez_lp_lab01.pdf
     |--- jperez_lp_lab01.tex

3 directories, 7 files
```

5. Calificación

Tabla 1: Rúbrica para contenido del Informe y evidencias

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Repositorio se pudo clonar y se evidencia la estructura adecuada para revisar los entregables. (Se descontará puntos por error o observación)	4	X	4	
2. Commits	Hay porciones de código fuente asociado a los commits planificados con explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	4	
3. Ejecución	Se incluyen comandos para ejecuciones y pruebas del código fuente explicadas gradualmente que permitirían replicar el proyecto. (Se descontará puntos por cada omisión)	4	X	4	
4. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos. (Se descontará puntos por error encontrado)	2	X	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente con explicaciones puntuales pero precisas, agregando diagramas generados a partir del código fuente y refleja un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4			
Total		20		16	

6. Referencias

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