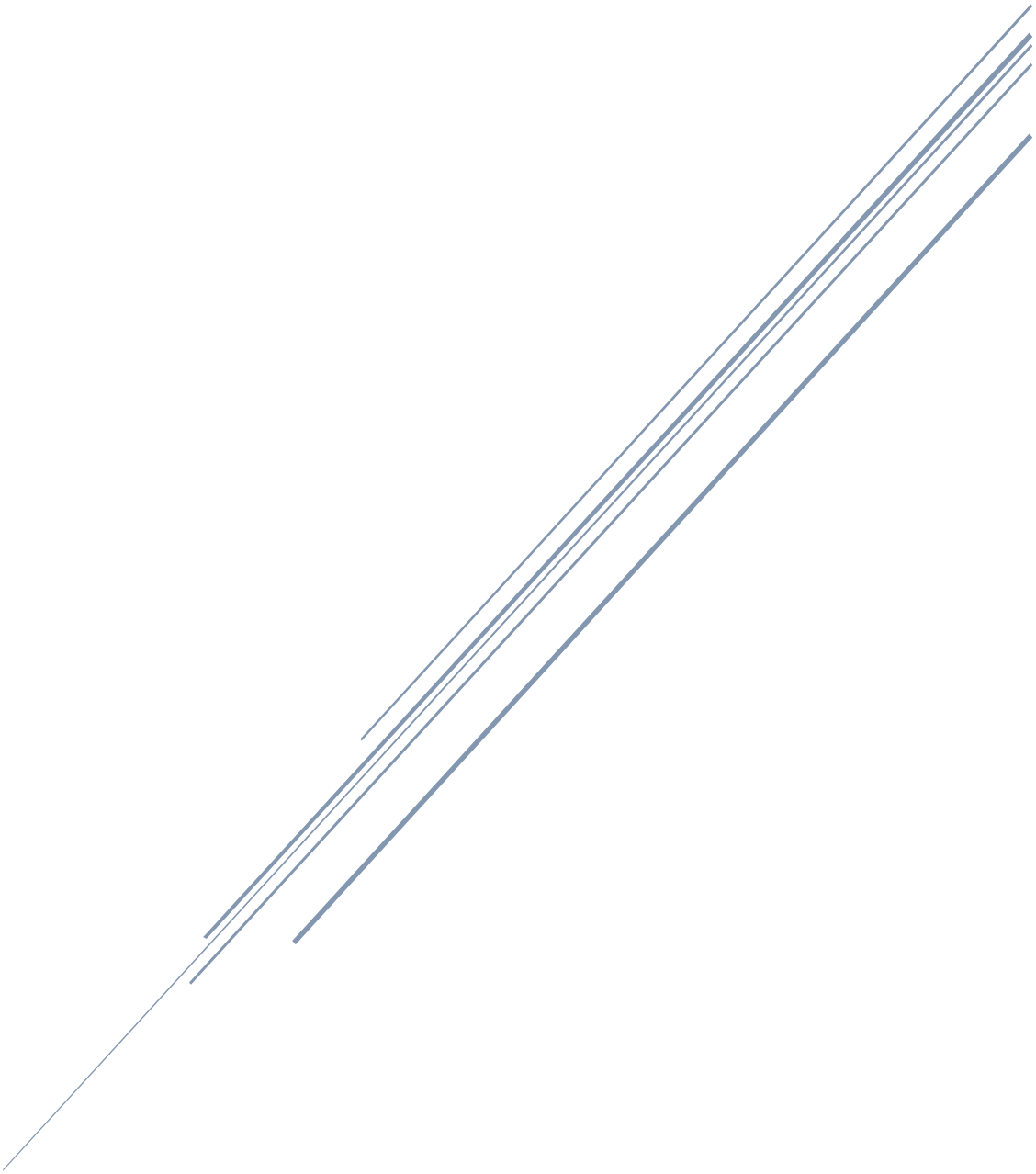
TRABAJO PRÁCTICO 0 – CURSO:XX

INDIVIDUAL

ALUMNO:



UTN FRBA

SINTAXIS Y SEMANTICA DE LOS LENGUAJES

Requisitos Generales para las Entregas de las Resoluciones

Cada trabajo tiene sus requisitos particulares de entrega de resoluciones, esta sección indica los requisitos generales, mientras que, cada trabajo define sus requisitos particulares.

Una resolución se considera entregada cuando cumple con los requisitos de tiempo y forma generales, aca descriptos, sumados a los particulares definidos en el enunciado de cada trabajo.

La entrega de cada resolución debe realizarse a través de GitHub, por eso, cada estudiante tiene poseer una cuenta en esta plataforma.

En el curso usamos repositorios GitHub. Uno público y personal y otro privado para del equipo.

Repositorios público y privado.

```
Usuario
`-- Repositorio público personal para la asignatura
Repositorio privado del equipo
```

Repositorio Personal para Trabajos Individuales

Cada estudiante debe crear un repositorio publico donde publicar las resoluciones de los trabajos individuales. El nombre del repositorio debe ser el de la asignatura. En la raíz del mismo debe publicarse un archivo readme.md que actúe como front page de la persona. El mismo debe estar escrito en notación Markdown y debe contener, como mínimo, la siguiente información:

- Sintaxis y semántica de los Lenguajes
- Curso.
- Ano de cursada, y cuatrimestre si corresponde.
- Legajo.
- Apellido.
- Nombre.

```
Usuario
`-- Repositorio público personal para la asignatura
    `-- readme.md // Front page del usuario
```

Repositorio de Equipo para Trabajos Grupales

A cada equipo se le asigna un **repositorio privado**. En la raíz del mismo debe publicarse un archivo readme.md que actúe como front page del equipo. El mismo debe estar escrito en notación Markdown y debe contener, como mínimo, la siguiente información:

- Sintaxis y semántica de los Lenguajes
- Curso.
- Ano de cursada, y cuatrimestre si corresponde.

- Numero de equipo.
- Nombre del equipo (opcional).
- Integrantes del equipo actualizados, ya que, durante el transcurso de la cursada el equipo puede cambiar:
  - Usuario GitHub.
  - Legajo.
  - Apellido.
  - Nombre.

Repositorio privado del equipo.

```
Repositorio privado del equipo
`-- readme.md // Front page del equipo.
```

Carpetas para cada resolución

La resolución de cada trabajo debe tener su propia carpeta, ya sea en el repositorio personal, si es un trabajo individual, o en el del equipo, si es un trabajo grupal. El nombre de la carpeta debe seguir el siguiente formato:

DosDigitosNumeroTrabajo-NombreTrabajo

Ejemplo 2.1. Nombre de carpeta

00-Hello

En los enunciados de cada trabajo, el número de trabajo para utilizar en el nombre de la carpeta esta generalizado con "DD", se debe reemplazar por los dos dígitos del trabajo establecidos en el curso.

Adicionalmente a los productos solicitados para la resolución de cada trabajo, la carpeta debe incluir su propio archivo readme.md que actúe como front page de la resolución El mismo debe estar escrito en notación Markdown y debe contener, como mínimo, la siguiente información:

- Numero de equipo.
- Nombre del equipo (opcional).
- Autores de la resolución:
  - Usuario github.
  - Legajo.
  - Apellido.
  - Nombre.
- Numero y titulo del trabajo.
- transcripción del enunciado.

- hipótesis de trabajo que surgen luego de leer el enunciado. Opcionalmente, para facilitar el desarrollo se recomienda incluir:
- un archivo .gitignore
- un archivo Makefile.
- archivos tests.

Carpeta de resolución de trabajo.

```
Carpeta de resolución de trabajo
|-- .gitignore
|-- Makefile
|-- readme.md // Front page de la resolución
`-- Archivos de resolución
```

Por ultimo, la carpeta **no debe incluir:**

- archivos ejecutables.
- archivos intermedios producto del proceso de compilación o similar.

Ejemplo de Estructura de Repositorios

Ejemplo completo.

```
usuario // Usuario GithHub
`-- Asignatura // Repositorio personal público para a la asignatura
    |-- readme.md // Front page del usuario
    |-- 00-Hello // Carperta de resolución de trabajo
    |    |-- .gitignore
    |    |-- readme.md // Front page de la resolución
    |    |-- Makefile
    |    |-- hello.cpp
    |    `-- output.txt
    `-- 01-Otro-trabajo
2019-051-02 // Repositorio privado del equipo
|-- redme.md // Front page del equipo
|-- 04-Stack // Carperta de resolución de trabajo
|    |-- .gitignore
|    |-- readme.md // Front page de la resolución
|    |-- Makefile
|    |-- StackTest.cpp
|    |-- Stack.h
|    |-- Stack.cpp
|    `-- StackApp.cpp
`-- 01-Otro-trabajo
```

Lenguaje de programación

En el curso se establece la versión del estándar del lenguaje de programación que debe utilizarse en la resolución.

TRABAJO N° 0

"Hello, World!" en C

Objetivos

- Demostrar capacidad para editar, compilar, y ejecutar programas C mediante el desarrollo de un programa simple.
- Tener un primer contacto con las herramientas necesarias para abordar la resolución de los trabajos posteriores.

Temas

- Sistema de control de versiones.
- Lenguaje de programación C.
- Proceso de compilación
- Pruebas.

Problema

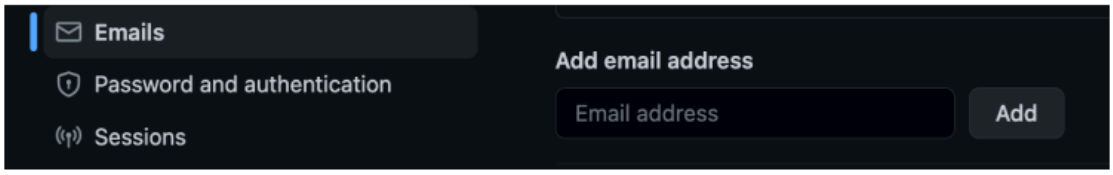
Adquirir y preparar los recursos necesarias para resolver los trabajos del curso.

Restricciones

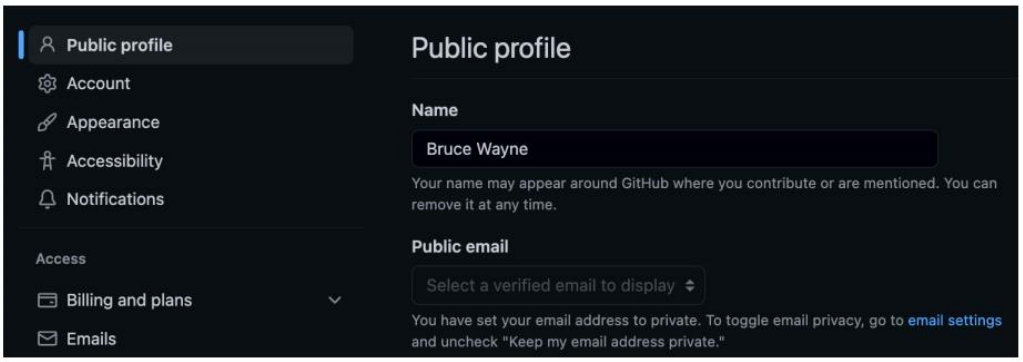
- Ninguna.

Tareas

1. Cuenta en GitHub
    a. Si no tiene, cree una cuenta GitHub.
    b. Si no lo hizo, asocie a su cuenta GitHub el email @frba y verifique. Es posible asociar mas de una cuenta email a una cuenta GitHub.



    c. Si no lo hizo, indique que su cuenta email @frba es pública. Esto permite a la catedra encontrar a los estudiantes. Si por temas de privacidad prefiere no tener como publica esa dirección, puede cambiarla al final del proceso.



2. Repositorio para público para la materia
    a. Cree un repositorio publico llamado SSL.
    b. En la raíz de ese repositorio, escriba el archivo readme.md que actúa como front page del repositorio personal.
    c. Cree la carpeta 00-CHelloworld.
    d. En esa carpeta, escriba un segundo archivo readme.md que actúa como front page de la resolución.
3. Compilador
    a. Seleccione, instale, y configure, y pruebe un compilador C11 o C18. Los mas osados pueden buscar un compilador que soporte C2x.
    b. Registre los resultados anteriores de la siguiente manera:
        i. Indique en el readme.md el compilador seleccionado.
        ii. Pruebe el compilador con un programa hello.c que envie a stdout la lmea Hello, World! o similar.
        iii. Ejecute el programa y verifique que la salida es la esperada.
        iv. Ejecute el programa con la salida redireccionada a un archivo output.txt; verifique su contenido.
4. Publicación
    a. Publique el trabajo en el repositorio personal SSL la carpeta 00- CHelloWorld con readme.md, hello.c, y output.txt.
5. Armado de Equipo.

Aunque el trabajo es individual, fomentamos la colaboración entre compañeros para su resolución. Consideramos que es una buena oportunidad para armar equipo para los trabajos siguientes que en su mayoría son grupales. El docente del curso indica la cantidad de integrantes mínima y máxima por equipo.

a. Informe el número de equipo en esta lista[1] .

   Con el número de equipo y cuenta @frba, la Catedra le envía la invitación al repositorio privado del equipo, por eso es importante que su cuenta GitHub tenga asociado como email público su email @frba, tal como indica el primer paso.

b. Luego de aceptar la invitación al repositorio privado del equipo, si lo desea, puede cambiar el email público en GitHub.

[1] https://docs.google.com/spreadsheets/d/19MZodiTIjD2WuImE8Y0WijNxIRdfL6vF_DvCn3uYlWg

3.6. Productos

```
Usuario                 // Usuario GitHub
`-- SSL                 // Repositorio público para la materia
    |-- readme.md       // Archivo front page del usuario
    `-- 00-CHelloWorld  // Carpeta el trabajo
        |-- readme.md   // Archivo front page del trabajo
        |-- hello.c     // Archivo fuente del programa
        `-- output.txt  // Archivo con la salida del programa
```

Referencia

• [Git101]

•[CompiladoresInstalacion]

• [KR1988] § 1.1 Comenzado