

Trabajos de Algoritmos y Esctructura de Datos

Esp. Ing. José María Sola, profesor.

Revision 0.1.0

Mar 2017

Table of Contents

1. Introducción	1
2. Requisitos Generales para las Entregas de las Resoluciones	3
2.1. Requisitos de Tiempo	3
2.2. Requisitos de Forma	3
2.2.1. Repositorio del Equipo en GitHub	3
2.2.2. Carpetas para cada Resolución	4
2.2.3. Header Comments (Comentarios Encabezado)	5
3. Problemas y Soluciones	7
4. Trabajo #0 — "Hello, World!" en C++	9
4.1. Objetivos	9
4.2. Entrega	9
4.3. Secuencia de Tareas	9
5. Trabajo #1 — Adición	11
5.1. Problema	11
5.2. Entregables	11
6. Trabajo #2 — Mayor de dos Números	13
6.1. Problema	13
6.2. Entregables	13

1

Introducción

Este documento presenta los enunciados de los trabajos prefijados del curso. Su resolución es grupal, salvo que se indique la contrario.

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, acá descriptos, sumados a los particulares descriptos en el enunciado de cada trabajo.

2.1. Requisitos de Tiempo

Cada trabajo establece la fecha y hora límite de entrega, los commits realizados luego de ese instante no son tomados en cuenta para la evaluación de la resolución del trabajo.

2.2. Requisitos de Forma

Requisitos de forma del repositorio, las carpetas de las resoluciones, y los encabezados de los archivos fuente.

2.2.1. Repositorio del Equipo en GitHub

La entrega de cada resolución debe realizarse a través de *GitHub*, para ello, a cada equipo se le asigna un **repositorio privado**. Lugo, el equipo debe crear una carpeta particular para cada resolución.

Front Page del Equipo

En la raíz del repositorio cada equipo debe diseñar un archivo `readme.md` que actúe como front page del equipo. Debe estar escrito en notación *Markdown* y debe tener, como mínimo, la siguiente información:

- Asignatura.
- Curso.
- Año de cursada, y cuatrimestre si corresponde.
- Número de equipo.
- Nombre del equipo (opcional).
- Integrantes del equipo actualizados, ya que, durante la transcurso de la cursada el equipo puede cambiar:
 - Usuario *GitHub*.
 - Legajo.
 - Apellido.
 - Nombre.

2.2.2. Carpetas para cada Resolución

La resolución de cada trabajo debe tener su propia carpeta en el repositorio del equipo.

Además de los productos solicitados por cada trabajo, la carpeta **sí debe incluir**:

- un archivo `readme.md` que actúe como front page de la resolución.

Para facilitar el desarrollo se **recomienda incluir**:

- un archivo `.gitignore`.
- un archivo `makefile`.
- archivos tests.

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

- archivos ejecutables.

- archivos intermedios producto del proceso de compilación o similar.

Nombre de la Carpeta

El nombre de la carpeta debe seguir el siguiente patrón:

ÚltimosTresDígitosDelCurso-DosDelEquipo-DosDígitosNúmeroTrabajo-
NombreTrabajo

Example 2.1. Nombre de carpeta

051-02-00-Hello

Front Page de la Resolución

Cada de resolución debe contar con un archivo `readme.md`, escrito en *Markdown* que contenga, como mínimo, la siguiente información:

- Número de equipo.
- Nombre del equipo (opcional).
- Autores:
 - Usuario github.
 - Legajo.
 - Apellido.
 - Nombre.
- Número y título del TP.
- Transcripción del enunciado.
- Hipótesis de trabajo que surgen luego de leer el enunciado.

2.2.3. Header Comments (Comentarios Encabezado)

Todo archivo fuente de debe comenzar con un comentario que indique el “Qué”, “Quiénes”, “Cuándo” :

```
/* Qué: Nombre
 * Breve descripción
 * Quiénes: Autores
 * Cuando: Fecha de última modificación
 */
```

Example 2.2. Header comments

```
/* Stack.h
 * Interface for a stack of ints
 * JMS
 * 20150920
 */
```

3

Problemas y Soluciones

Todos los archivos `readme.md` que actúan como *Front Page* de la resolución, deben contener una descripción del *análisis del problema* y una *síntesis de la solución*.

Luego del enunciado y las hipótesis, cada `readme.md` debe contener:

- Etapa de Análisis:
 - Entradas: nombres y tipos de datos.
 - Proceso: nombre descriptivo
 - Salidas: nombres y tipos de datos.
- Etapa de Síntesis:
 - Diseño del Algoritmo:
 - Léxico del Algoritmo.
 - Representación visual o textual del Algoritmo.
 - Implementación del algoritmo en C++.

4

Trabajo #0 — "Hello, World!" en C++

Este es un trabajo individual.

4.1. Objetivos

- Demostrar con, un programa simple, que se está en capacidad de editar, compilar, y ejecutar un programa C++.
- Contar con las herramientas necesarias para la resolución de los trabajos.

4.2. Entrega

- A las 13:00 del día de la segunda clase.

4.3. Secuencia de Tareas

1. Solicitar inscripción al Grupo Yahoo.
2. Registrarse en GitHub
3. Crear un repositorio público con el nombre `cppHelloWorld`.
4. Seleccionar, instalar, y configurar un compilador C++14.
5. Probar el compilador con un programa `hello.cpp` que envíe a `cout` la línea `hello, world!` o similar.
6. Ejecutar el programa, y capturar su salida en un archivo de texto `output.txt`.
7. Publicar `hello.cpp` y `output.txt` en GitHub.
8. Enviar a UTNFRBAAED@yahoogroups.com¹ usuario y repositorio GitHub.

¹ <mailto:UTNFRBAAED@yahoogroups.com>

5

Trabajo #1 — Adición

5.1. Problema

Escribir los pasos para mostrar la suma de dos números que ingresa el usuario.

5.2. Entregables

- Carpeta: Adición
- Implementación: Adición.cpp.

6

Trabajo #2 — Mayor de dos Números

6.1. Problema

Dado dos números informar cuál es el mayor.

6.2. Entregables

- Carpeta Mayor
- Implementación: Mayor.cpp.

