**Algoritmos y Estructura de datos III - Año 2025 – 2do Periodo**
**Prof. Cristian Cappo / Prof. Luis Moré**

# Tarea 1 – U1 (Java)

Fecha Entrega Límite 17/08/2025 23:59 hs.
TP = 40pts

---

Cada archivo fuente deberá tener la identificación del grupo (g##) y los integrantes del grupo con la indicación del número de tarea. Por cada integrante colocar: apellido, nombre, CIC y sección.

De incluir la Declaración de Honor con el texto mostrado en la primera clase.

**Ejercicio 1 (20p)**

Una funcionalidad habitual de los navegadores es la opción de moverse hacia adelante o atrás de las páginas web según el orden en que fueron visitadas.

Este problema consiste en implementar dicha funcionalidad haciendo un pequeño simulador en Java. Para ello su simulador debe soportar las siguientes operaciones:

| Comando | Operación | Descripción |
|---|---|---|
| A | ANTERIOR | (back) coloca como actual la que estaba anteriormente al actual. Si no existe un página anterior entonces el comando se ignora y se imprime el mensaje “IGNORADO” |
| S | SIGUIENTE | (forward) coloca como página actual la que fue visitada luego de la página actual. Si no existe página siguiente el comando se ignora y se imprime “IGNORADO” |
| U <URL> | VISITAR <URL> | Ingresar el URL a visitar en <URL>. Obviamente, en este caso ya no existirían páginas hacia adelante. Se imprime el mismo <URL> |
| L | LISTAR URLs | Se listan las URLS en el orden de visita, del más nuevo al más antiguo. Debe indicar cual es la URL actual (indicador <-- ) |
| Q | SALIR | Salir del simulador del navegador |

Al iniciar el simulador se espera el ingreso de uno de los posibles comandos (A,S,U,L,Q). Se debe empezar el simulador con una URL que se pasa como parámetro de la línea de comandos. Aceptar los comandos en mayúsculas o minúsculas. Siempre se imprime la URL actual y eventualmente los mensajes o el listado.

Abajo se muestra una interacción con el simulador:

```
$ java minibrow  http://www.pol.una.py/
> Ingrese comando (A,S,U,L,Q) (URL actual = http://www.pol.una.py/): U http://www.ibm.com/
> Ingrese comando (A,S,U,L,Q) (URL actual = http://www.ibm.com/): L
>> -----
>> http://www.ibm.com/ <--
>> http://www.pol.una.py/
>> -----
> Ingrese comando (A,S,U,L,Q) (URL actual = http://www.ibm.com/): S
>> IGNORADO
> Ingrese comando (A,S,U,L,Q) (URL actual = http://www.ibm.com/): A
> Ingrese comando (A,S,U,L,Q) (URL actual = http://www.pol.una.py/):
```

**Observaciones:**

- La solución debe alzar en el sitio indicado en el aula virtual.
- Debe incluir los comentarios en los archivos fuentes.
- Se considerará el orden (espaciado, indentación, comentarios, etc), estilo (nombre de variables y funciones, comentario en los métodos y clases, etc) y datos (lo requerido).
- Utilizar al menos un TAD básico para implementar la solución.
- Se tendrá en cuenta el correcto uso de las estructuras de datos y de la aplicación de los conceptos de TAD.
- NO puede usar ninguna estructura de datos del API de Java, debe implementar todas las que necesite (esto

invalidará su ejercicio).

A continuación se muestra un ejemplo:

**Ejecución del programa(asume que la clase principal se llama Minibrow )**

```
$ java Minibrow https://www.pol.una.py
```

| Comando | URL actual | Se imprime |
|---|---|---|
| | https://www.pol.una.py | |
| U https://www.google.com/ | https://www.google.com/ | |
| U https://www.facebook.com/ | https://www.facebook.com/ | |
| A | https://www.google.com/ | |
| A | https://www.pol.una.py/ | |
| A | https://www.pol.una.py/ | IGNORADO |
| S | https://www.google.com/ | |
| U https://www.ibm.com/ | https://www.ibm.com/ | |
| A | https://www.google.com/ | |
| A | https://www.pol.una.py/ | |
| S | https://www.google.com/ | |
| L | https://www.google.com/ | ---------- |
| | | https://www.ibm.com/ |
| | | https://www.google.com/ <-- |
| | | https://www.pol.una.py/ |
| | | ------ |
| S | https://www.ibm.com/ | |
| S | https://www.ibm.com/ | IGNORADO |
| L | https://www.ibm.com/ | ---------- |
| | | https://www.ibm.com/ <-- |
| | | https://www.google.com/ |
| | | https://www.pol.una.py/ |
| U https://www.oracle.com | https://www.oracle.com/ | ------ |
| L | https://www.oracle.com/ | |
| | | ------ |
| | | https://www.oracle.com <-- |
| | | https://www.ibm.com/ |
| | | https://www.google.com/ |
| | | https://www.pol.una.py/ |
| Q | | ------ |

| Rúbrica | Puntaje |
|---|---|
| Uso de ED adecuada para la solución al problema planteado | 2 |
| Solución correcta con la funcionalidad esperada (3 pt por funcionalidad) | 12 |
| Definición de ED básico (Lista, Pila, Cola) para uso en la solución | 4 |
| Orden, estilo, comentarios y datos | 1 |
| Utilización correcta de Excepciones | 1 |

**Ejercicio 2 (20p)**

Suponga que tiene un arreglo de N elementos y necesita determinar el elemento mayoritario. Un elemento mayoritario de un arreglo A[1..n] es aquel que se repite más de la mitad de veces del tamaño del arreglo.

Restricción: los elementos del arreglo no son precisamente de un tipo ordenable (como los enteros). Esto es, sólo puede comparar los elementos por igualdad (*equals*). Así, los elementos pueden ser por ejemplo: audio, imagen, documento, flujo de red (stream), cadenas, etc. Desarrolle su solución con un método genérico que retorne el elemento mayoritario si lo hay o null si no lo hay.

En el método main:

Pruebe elementos aleatorios numéricos donde 100.000 <= n <= 1.000.000. Genere una tabla de ejecución del algoritmo para varios valores de n. Utilice incrementos de 100.000. Imprima la cantidad de tiempo que llevó (en ms.) sin tener en cuenta la generación de números aleatorios. ¿Existe una relación entre el número n y el tiempo que lleva la ejecución del algoritmo? Comente en el propio código.

Su tabla debe lucir como sigue:

```
N           Tiempo (en ms)
--          -------
100000      ###,###
200000      ###,###
..          ..
1000000     ###,###
```

**Observaciones:**

- Debe alzar en el sitio indicado en el aula virtual.
- Debe incluir los comentarios en los archivos fuentes.
- Se considerará el orden (espaciado, indentación, comentarios, etc), estilo (nombre de variables y funciones, comentario en los métodos y clases, etc) y datos (lo requerido).

| **Rúbrica** | **Puntaje** |
|---|---|
| Desarrollo de la función genérica | 1 |
| Funcionamiento correcto del método mayoritario respetando la comparación de objetos. | 8 |
| Desarrollo correcto del método main de acuerdo a la especificación ( explicación 2p) | 8 |
| Orden, estilo, comentarios y datos | 1 |
| Uso correcto de excepciones | 2 |