UNIVERSIDAD ORT
Uruguay

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

EVALUACIÓN	Obligatorio	**GRUPO**	Todos	**FECHA**	21/3
MATERIA	Estructura de Datos y Algoritmos 2				
CARRERA	Ingeniería en Sistemas				
CONDICIONES	- Puntos: Máximo: 30 Mínimo: 1 LA ENTREGA SE REALIZA EN FORMA ONLINE EN ARCHIVO NO MAYOR A 40 MB EN FORMATO ZIP, RAR O PDF. **IMPORTANTE:** - Inscribirse - Formar grupos de hasta dos personas. - Subir el trabajo a Gestión antes de la hora indicada, ver hoja al final del documento: “RECORDATORIO”				

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Obligatorio

El obligatorio de Estructuras de Datos y Algoritmos 2, para el semestre 1 de 2023, está compuesto por un conjunto de **10** problemas a resolver.

El desarrollo del obligatorio:

- Se deberá realizar en grupos de **2 estudiantes**.
- Se podrá utilizar tanto C++ (C++11) como Java (jdk8).
  Para asegurar que utilizan C++11 deberán tener instalado el compilador C++ de GNU e invocar la siguiente orden: **`g++ -std=c++11 ...`**
- No se puede usar ninguna librería, clase, estructura, función o variable no definida por el estudiante, a excepción de <iostream>, <string> y <cstring> para C++. y System.in, System.out y String para Java.
- Si se quiere usar algo que no sea lo listado anteriormente, consultar previamente al profesor.
- Se deberá utilizar el formato de ejecución trabajado durante el curso:
  `miSolucion < input > output`
- La cátedra proveerá un conjunto de casos de prueba, pares de entrada y salida esperada. Se deberá comparar la salida de la solución contra una salida esperada.
- **De no cumplir las condiciones anteriores el ejercicio se considera invalido, por lo tanto 0 puntos.**

La entrega:

- El formato de entrega se encuentra definido en este enlace.

La corrección:

- La corrección implica la verificación contra la salida esperada, así también como la corrección del código fuente para verificar que se cumplan los requerimientos solicitados (órdenes de tiempo de ejecución, usos de estructuras o algoritmos en particular, etc.).
- Se verificarán TODOS los casos de prueba. Si el programa no termina para un caso de prueba, puede implicar la pérdida de puntos.
- Si el código fuente se encuentra en un estado que dificulta la comprensión, podrá perder puntos.
- Los códigos dados en clase como algoritmos encontrados de otras fuentes serán usados como referencia. **Considerándose copia el uso de ellos.**
- **Se utilizará MOSS para detectar copias.**

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

La defensa:

- Luego de la entrega, se realizará una defensa de autoría a cada estudiante **de manera individual**.

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

1. Base de Datos de Estudiantes

Nombre de archivo: ejercicio1.cpp/Ejercicio1.java

Letra

Se tiene una base de datos de estudiantes de la Universidad ORT. En esta instancia, cada estudiante está representado por una palabra conformada por las iniciales de su nombre y sus dos apellidos completos. Se conoce al conjunto de notas de exoneraciones y exámenes en diferentes asignaturas. Se desea implementar una tabla de hash cerrado que permita buscar los estudiantes por número de identificación y obtener el promedio de sus notas.

Se pide implementar una función en que reciba como entrada los códigos de identificación de los estudiantes y sus respectivas notas de exoneración o aprobación de examen. La función deberá imprimir por pantalla el código de identificación de los alumnos y el PAC (Promedio Acumulado de Calificaciones) de cada uno de ellos. El listado deberá respetar el orden en el que fueron introducidos los alumnos y sus notas. La implementación debe utilizar una tabla de hash cerrado con resolución de colisiones distinta al método de exploración lineal.

Input

La primera línea de la entrada contiene un número entero positivo A, el número de estudiantes de la universidad. Le siguen A líneas distintas de formato N K K_1 K_2 … K_k siendo N las iniciales y apellidos del estudiante, K la cantidad de aprobaciones y exoneraciones y los siguientes valores las notas obtenidas.

Salida

A líneas de formato N P, siendo N las iniciales y apellidos del estudiante y P el PAC del alumno.

Restricciones

- Utilizar una tabla de **hash cerrado**.
- Resolver en orden temporal: O(A) promedio, siendo A la cantidad de usuarios.

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Ejemplos de entrada y salida

Entrada

```
3
CHBuenoSuarez 4 10 30 50 70
DMForlanCorazo 3 100 100 100
FSValverdeDipetta 5 100 90 80 70 60
```

Salida

```
CHBuenoSuarez 40
DMForlanCorazo 100
FSValverdeDipetta 80
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

2. Libro

Nombre de archivo: ejercicio2.cpp/Ejercicio2.java

Letra

Se tiene un libro, el cual tiene una cantidad N de palabras, siendo en total P palabras distintas. Se desea implementar una función en que permita mostrar las distintas P palabras ordenadas y su cantidad de apariciones en el libro.

La función deberá recibir como entrada el número P de palabras, seguido por una palabra por línea, totalizando P+1 líneas. Debe mostrar por pantalla las P palabras distintas ordenadas alfabéticamente de forma descendente, con cada línea conteniendo la palabra y sus apariciones en el texto.

Input

La primera línea de la entrada contiene un número entero positivo N, seguido por N líneas con una palabra cada una.

Salida

Imprime P líneas conteniendo una palabra y su cantidad de apariciones cada una. Las P líneas deben estar ordenadas alfabéticamente de forma descendente por palabra.

Restricciones

- La operación debe tener orden O(P * Log_2 P) peor caso, siendo P la cantidad de palabras distintas en el texto.

Ejemplos de entrada y salida

Entrada

```
10
sos
mi
locura
y
mi
pasion
tuya
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

```
es
mi
vida
```

Salida

```
y 1
vida 1
tuya 1
sos 1
pasion 1
mi 3
locura 1
es 1
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

3. Aerolínea **Nombre de archivo:** ejercicio3.cpp/Ejercicio3.java

Letra

Una aerolínea desea optimizar su sistema de check-in para favorecer a los pasajeros más frecuentes de la misma. Se recibirá un número R de reservas distintas, compuestas por P personas indicadas en la reserva. Cada persona tiene un pasaporte y cantidad de vuelos realizados con la aerolínea. Las personas pertenecientes a la misma reserva ingresan juntos, favoreciendo inicialmente a aquellas reservas que tienen un más alto promedio de vuelos realizados con la aerolínea.

Input

La función deberá recibir como entrada el número R de reservas, seguido por R grupos conformados por un número P de personas en la reserva para seguir con P líneas, cada una conteniendo al número de pasaporte del viajero y la cantidad de vuelos realizados.

Salida

Debe mostrar por pantalla R líneas ordenadas descendientemente por promedio de vuelos de los integrantes de la reserva. Las líneas deben contener el promedio de vuelos de los integrantes seguido por sus pasaportes, todos separados por un espacio y en el mismo orden original en el que fueron ingresados.

Restricciones

- El orden de ejecución de la carga de datos no puede superar O(P + R) caso promedio y O(P + R * Log_2 R) peor caso, mientras que la impresión no debe llevar más de O(R * Log_2 R) peor caso.

Ejemplos de entrada y salida

Entrada

```
3
2
A123456 10
B234567 11
1
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

```
C345678 40
3
D456789 50
E567890 10
F678901 30
```

Salida

```
40 C345678
30 D456789 E567890 F678901
10.5 A123456 B234567
```

UNIVERSIDAD ORT
Uruguay

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Formato de entrada de ejercicios de grafos

Todos los ejercicios de grafos tendrán la misma codificación para los grafos. Es decir, una parte del formato de entrada, la que corresponde a la información del grafo será siempre igual. A continuación se describe:

```
V
E
v1 w1 [c1]
v2 w2 [c2]
...
vi wi [ci]
...
vE wE [cE]
```

Cada grafo comienza con la cantidad de vértices, V. Los vértices siempre serán números, a menos que se especifique lo contrario. Por ejemplo, si $V=3$, entonces los vértices serán: $\{1, 2, 3\}$ (siempre serán numerados a partir de 1).

La siguiente línea corresponde a la cantidad de aristas, `E`. Las siguientes `E` líneas en el formato `v w c` corresponden a las aristas (v,w) con costo c si el grafo *es ponderado*, o en el formato `v w`, correspondiente a la arista (v,w) si *no es ponderado*. Es decir, c es opcional (`[c]`).

El grafo será dirigido o no, dependiendo el problema en particular. En caso de ser *no dirigido* solo solo se pasará un sentido de la arista, es decir, (v,w) pero no (w,v) (queda implícito). Por ejemplo:

```
2
1
1 2
```

Representa al grafo completo de dos vértices y aristas: $\{(1,2), (2,1)\}$.

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

4. Montevideo

Nombre de archivo: ejercicio4.cpp/Ejercicio4.java

Letra

Se tiene una red de ciudades, cada una de las cuales está conectada por carreteras de simple o doble sentido. La red contiene N ciudades numeradas del 1 al N, y M carreteras. La distancia entre cada par de ciudades está dada por un valor numérico entero positivo. Algunas de las carreteras y ciudades pueden estar deshabilitadas temporalmente debido a mantenimiento, daño, etc. Se desea implementar una función en que permita determinar la distancia mínima entre dos ciudades dadas de la red, tomando en cuenta estas carreteras y ciudades deshabilitadas.

La función deberá recibir como entrada el número de ciudades N, seguido por el número de carreteras M. Luego, recibirá M líneas con cinco enteros cada una: los números de las dos ciudades conectadas por la carretera, la distancia entre ellas, un número que indica si la calle es de sentido simple (1) o doble (2) y un número que indica si la carretera está habilitada (1) o deshabilitada (0). Finalmente, recibirá dos enteros C1 y C2, que representan las ciudades de origen y destino respectivamente. La función deberá mostrar por pantalla en una línea la distancia mínima entre las ciudades C1 y C2 (tomando en cuenta las carreteras y ciudades deshabilitadas) y en otra el camino entre ambas en formato array.

Input

La primera línea de la entrada contiene dos números enteros positivos separados por un espacio, N y M. Las siguientes M líneas contienen cuatro enteros cada una, separados por espacios, que representan los vértices que conecta la carretera, su distancia y si está habilitada (1) o deshabilitada (0). La siguiente línea contiene un número entero positivo K, el número de ciudades deshabilitadas. Las siguientes K líneas contienen un número entero que representa la ciudad deshabilitada. Luego indica un número L seguido por L líneas con cada una indicando 2 números de ciudades que tendrán a la carretera que los une deshabilitada. La última línea contiene dos enteros C1 y C2 representando a la ciudad origen y destino.

Salida

Debe mostrar por pantalla un número entero que representa la distancia mínima entre las ciudades C1 y C2, seguido por una línea en formato array mostrando las ciudades que forman parte del camino. En caso que no exista camino válido entre el origen y el destino, deberá mostrar -1 y luego una línea con un array vacío ([])

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Restricciones

- Debe implementarse con un grafo implementado con Listas de Adyacencia, ya que el grafo es disperso
- El orden de ejecución no puede superar $O(M + N \text{ Log}_2 N)$

Ejemplos de entrada y salida

Entrada

```
5 7
1 2 5 2 1
1 4 6 1 1
1 5 8 1 0
2 3 8 2 0
3 4 3 1 1
3 5 1 2 1
5 4 3 2 1
1
2
1
1 5
1 5
```

Salida

```
9
[1, 4, 5]
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

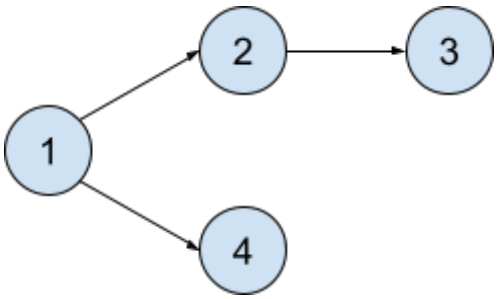
5. ORDEN TOP.

Nombre de archivo: ejercicio5.cpp/Ejercicio5.java

Letra

Implementar un algoritmo que dado un grafo dirigido, no ponderado, débilmente conexo, disperso y acíclico devuelva el orden topológico.

Ante la posibilidad de varios órdenes topológicos, desarrollar la estrategia que se describe a continuación. Supongamos el siguiente grafo:



Existen tres órdenes, a saber: <1,2,3,4>, <1,4,2,3> y <1,2,4,3>. Nuestro algoritmo debe retornar el último. La estrategia consiste en las siguientes dos condiciones:

1. Ante la posibilidad de dos o más órdenes, siempre retornar el que contenga más cantidad de nodos “consecutivos en el mismo nivel”. Con esto último queremos decir que tanto <1,2,4,3> como <1,4,2,3> mantienen a 2 y 4 consecutivos, mientras que <1,2,3,4> no. Decimos que 2 y 4 pertenecen la mismo nivel porque se encuentra a la misma cantidad de saltos desde el primer vértice en el orden[1].
2. En caso de que existan varios órdenes luego de aplicar la condición anterior, elegir el que sitúe a los nodos consecutivos en orden ascendente. En el ejemplo, el orden final será <1,2,4,3>.

El algoritmo debe ejecutarse en O(E + V log V) tiempo de ejecución.

[1] Notar que si el grafo tuviera varios vértices iniciales, i.e, de grado de incidencia cero, esto no afectará la determinación de la condición o restricción.

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Input

(ver entrada de grafo)

Salida

La salida contendrá V líneas, donde cada línea en la posición i representa al vértice v_i que se encuentra en la posición i del orden topológico.

6. Sin Pareja

Nombre de archivo: ejercicio6.cpp/Ejercicio6.java

Letra

Dado un array de **números ordenados**, debe encontrar aquel número que solo aparece una vez (los demás aparecen dos veces).

arr = [2,2,5,5,7,7,10,10,**11**,14,14,16,16] -> devuelve **11**

Restricciones:

- $O(LgN)$ temporal (siendo N el tamaño del array)
- $O(1)$ espacial (sin tener en cuenta el array)
- Debe realizarse con la táctica **dividir y conquistar (decrease and conquer)**

Formato de entrada

```
N
n1
n2
…
nN
```

La primera línea (N) es la cantidad de elementos del array y las siguientes N líneas son los elementos del array.

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Formato de salida

Contendrá 1 sola línea con el número sin pareja.

7. A ÚLTIMO MOMENTO

Nom. de archivo: ejercicio7.cpp/Ejercicio7.java

Letra

El obligatorio de Estructuras de datos y algoritmos 3 es muy similar al de su materia antecesora, con la pequeña diferencia de que no todos los ejercicios tienen el mismo puntaje.

Un día antes de la entrega, el profesor le recuerda que la entrega de todos los archivos no debe superar los S MB (megabytes) ni las L líneas de código entre todos ellos.

Como usted olvidó este detalle y no tiene tiempo a refactorizar su código para que cumpla con las restricciones, decide elegir aquellos ejercicios/archivos que le garanticen un mejor puntaje.

De cada ejercicio/archivo, usted sabe su tamaño, cantidad de líneas de código y el puntaje.

Restricciones:

- $O(N*S*L)$ temporal y espacial
- Debe realizarse con la táctica de **programación dinámica**.

Formato de entrada

```
N
S
L
t1 l1 p1
t2 l2 p2
…
tN lN pN
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

La primera línea (N) es la cantidad de archivos, la segunda (S) y tercera (L) es la cantidad máxima de tamaño y líneas de código respectivamente para la entrega. Por último siguen N líneas con la información de tamaño, líneas de código y puntaje de cada archivo/entrega.

Salida

La cantidad de puntos conseguidos (una sola linea).

8. Aeropuerto

Nom. de archivo: ejercicio8.cpp/Ejercicio8.java

Letra

Se está construyendo un nuevo aeropuerto y no se sabe cuántas plataformas se deben construir. Lo que se sabe son los horarios de llegada y salida de cada avión. Se sabe además, que en cada plataforma solo puede haber un avión a la vez y que se quiere saber la cantidad mínima de plataformas para poder operar.

Restricciones:

- O(NLgN) temporal (siendo N la cantidad de vuelos)
- O(N) espacial (siendo N la cantidad de vuelos)
- Debe realizarse con la táctica **greedy**

Formato de entrada

```
N
ll1 s1
ll2 s2
...
llN sN
```

La primera línea (N) es la cantidad de elementos de vuelos, las siguientes N líneas es la información de horarios de llegada y partida de cada avion.

Formato de salida

Contendrá 1 sola línea con la cantidad mínima de líneas para poder operar.

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Nota: Los horarios no tienen ningún orden en particular.

9. COMBINATORIA

Nom. de arch.: ejercicio9.cpp/Ejercicio9.java

Letra

Dados dos números, N y K, se pide retornar el resultado de calcular $C(N,K)$ o $\binom{N}{K}$, es decir, la cantidad de maneras de tomar N elementos en grupos de K sin repetir, con $0 \leq K \leq N$.

Se pide resolver en tiempo $O(NK)$ y espacio $O(N)$.

Por ejemplo, $C(5,2)=10$. Corresponde a los siguientes conjuntos: $|\{1,2\}, \{1,3\}, \{1,4\}, \{1,5\}, \{2,3\}, \{2,4\}, \{2,5\}, \{3,4\}, \{3,5\}, \{4,5\}|=10$.

Formato de entrada

```
N
P
N1 K1
N2 K2
...
NP KP
```

La primera línea indica las cota N para los siguientes P casos, es decir, $N_i \leq N$ y $K_i \leq N$ para todo i con $1 \leq i \leq P$. La segunda indica la cantidad de casos de prueba, P. Las siguientes P líneas contienen cada caso dado por dos números N_i y K_i con $1 \leq i \leq P$ y $0 \leq K_i \leq N_i \leq N$ *ordenados* según el par (N_i,K_i) para los cuales se desea calcular la cantidad de combinaciones $C(N_i,K_i)$.

Formato de salida

```
C(N1,K1)
C(N2,K2)
...
C(NP,KP)
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

La salida contendrá P líneas, cada una indicando la cantidad de combinaciones $C(N_i,K_i)$ con $1 \leq i \leq P$.

10. LABERINTO

Nom. de archivo: ejercicio10.cpp/Ejercicio10.java

Letra

Es el primer día de retorno a clases, y resulta que se ha olvidado de la disposición de la estructura de la facultad. Si visualizamos la facultad como una matriz MxN, con 1≤x≤M y 1≤y≤N, podemos distinguir en los casilleros una letra que puede corresponder a paredes (P), bedelía (B) y pasillo/salón (C). Para saber cuál es el salón, primero debe pasar por bedelías para consultar cuál le corresponde, luego tiene que dirigirse al salón evitando paredes y otros salones.
Luego de subir una serie de pisos se encuentra exhausto, y para caminar lo menos posible, desea buscar el camino óptimo (más corto) para llegar al salón destino habiendo pasado primero por bedelías. Ten en cuenta que sólo se puede mover hacia arriba, abajo, izquierda o derecha (no se permiten movimientos diagonales).

Por ejemplo, dado el siguiente piso representado en una matriz 5x5 con (1, 4) de origen y (5, 4) de destino, la solución se encuentra coloreada:

P	P	C	P	C
C	C	C	P	P
C	P	C	C	C
C	C	B	P	C
P	C	C	C	C

Formato de entrada

```
M N
L1,1 L1,1 L2,1 ... LM,1
L1,2 L1,2 L2,2 ... LM,2
L1,3 L1,3 L2,3 ... LM,3
...
```

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

$L_{1,N}$ $L_{1,N}$ $L_{2,N}$ ... $L_{M,N}$
P
$x_{i,1}$ $y_{i,1}$ $x_{f,1}$ $y_{f,1}$
$x_{i,2}$ $y_{i,2}$ $x_{f,2}$ $y_{f,2}$
...
$x_{i,P}$ $y_{i,P}$ $x_{f,P}$ $y_{f,P}$

Los primeros dos números indican las dimensiones de la matriz. Los siguientes MxN números representan a la matriz como se indica en la tabla. Luego sigue P indicando la cantidad de casos de prueba. Por último, siguen 4P números representando las ordenadas y coordenadas de los puntos iniciales y finales, respectivamente, en el orden indicado.

Formato de salida

s_1
s_2
s_3
...
s_p

El archivo de salida deberá contener la cantidad de casilleros que movimientos que tuvo que hacer para llegar al destino (origen y destino inclusive) o 0 si no existe.

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

RECORDATORIO: IMPORTANTE PARA LA ENTREGA

Obligatorios (Cap.IV.1, Doc. 220)

La entrega de los obligatorios será en formato digital online, a excepción de algunas materias que se entregarán en Bedelía y en ese caso recibirá información específica en el dictado de la misma.

Los principales aspectos a destacar sobre la **entrega online de obligatorios** son:

1. La entrega se realizará desde **gestion.ort.edu.uy**
2. Previo a la conformación de grupos cada estudiante deberá estar inscripto a la evaluación. S**ugerimos realizarlo con anticipación.**
3. **Uno de los integrantes del grupo de obligatorio será el administrador del mismo** y es quien formará el equipo y subirá la entrega
4. Cada equipo debe entregar **un único archivo en formato zip o rar** (los documentos de texto deben ser pdf, y deben ir dentro del zip o rar)
5. El archivo a subir debe tener **un tamaño máximo de 40 mb**
6. Les sugerimos **realicen una 'prueba de subida' al menos un día antes**, donde conformarán el **'grupo de obligatorio'.**
7. La **hora tope para subir el archivo será las 21:00** del día fijado para la entrega.
8. La entrega se podrá realizar desde cualquier lugar (ej. hogar del estudiante, laboratorios de la Universidad, etc)
9. Aquellos de ustedes que presenten alguna dificultad con su inscripción o tengan inconvenientes técnicos, por favor pasar por la oficina del Coordinador o por Coordinación adjunta **antes de las 20:00hs.** del día de la entrega

Si tuvieras una situación particular de fuerza mayor, debes dirigirte con suficiente antelación al plazo de entrega, al Coordinador de Cursos o Secretario Docente.