Pautas generales para los trabajos prácticos
Problemas, Algoritmos y Programación

Estas pautas son válidas para todos los trabajos prácticos de la materia. Todo el contenido de este documento se considera parte del enunciado de cada trabajo práctico.

Sobre la conformación de grupos

Los grupos deberán estar conformados por 4 integrantes, permitiendo un máximo de 3 grupos de 5 integrantes. La cantidad de grupos de 5 integrantes será el resto de dividir la cantidad de alumnos por 4, teniendo prioridad para conformar grupos de 5 personas los primeros que informen los integrantes de su grupo de 5, aunque nadie tiene garantizado poder formar un grupo de 5 integrantes hasta la fecha límite en la que se armen los grupos de TPs.

Los integrantes de cada grupo deberán ser notificados por correo electrónico a `ltaravilse+pap@gmail.com` antes de finalizado el viernes 26 de agosto. Quienes no tengan grupo para esta fecha o tengan un grupo incompleto (2 o 3 personas) pueden avisar, a la misma dirección, que tienen intención de hacer la materia, para que se les asigne un grupo si llegada esta fecha no lo tienen todavía. Quienes no avisen que van a hacer la materia para el viernes 26 de agosto quedarán afuera de la materia.

Si algún integrante de algún grupo deja la materia de un trabajo práctico a otro, los demás integrantes del grupo deberán avisar de esto a los docentes de la materia para que, en caso de haber muchos grupos de 2 o 3 integrantes, se puedan rearmar los grupos. Llegado el caso de que muchas personas dejen la materia, todas de distintos grupos, y queden muchos grupos de 3 integrantes, para intentar no romper grupos ya armados de 3 personas, y para que no haya grupos con cantidad desproporcionada de integrantes, se les pedirá a los grupos de 5 personas que elijan uno de sus integrantes para sumarse a un grupo de 3. Esto sólo sucederá si hay una cantidad importante de grupos que queden de 3 personas.

Sobre lo que se espera de un trabajo práctico

Los trabajos prácticos de la materia estarán fundamentalmente enfocados a la parte de programación. Lo más importante es poder escribir un código que resuelva cada problema, que resuelva el mismo utilizando los algoritmos y estructuras de datos vistos en la materia, y respete las complejidades pedidas.

Si bien siempre se pide un código que sea genérico y pueda resolver cualquier instancia del problema sin importar el tamaño de la misma, en la vida real existen restricciones de memoria (para el tamaño de los arreglos por ejemplo) o sobre el rango de números representables (hay tipos de datos que no pueden representar enteros de más de 32 bits, 64 bits, con signo, sin signo, etc). Para esto, el enunciado de cada problema tendrá cotas recomendadas para testear. Todo caso de test que cumpla con estas cotas deberá poder ser resuelto con el código que presenten del problema, teniendo en cuenta estas restricciones. Para todos los demás casos (cuando no se cumplan estas cotas), el código deberá ser capaz de resolverlo si no consideramos los números representables, la memoria disponible, y todas las demás restricciones que sean propias de la implementación, y no restricciones teóricas del algoritmo utilizado. Incluso estas instancias deben poder ser resueltas por el código provisto cumpliendo la complejidad pedida.

Con respecto al informe se espera una explicación que demuestre un entendimiento de:

- Qué hace el código y qué problema abstracto está resolviendo.
- Por qué el enunciado pedido se puede modelar como el problema abstracto del item anterior.
- Por qué la resolución que pensaron del problema abstracto es correcta.
- Qué hipótesis se asumen sobre la entrada del problema y por qué se cumplen en el enunciado.
- En caso de utilizar un algoritmo visto en la materia, cómo este algoritmo se integra con el resto del código, y cuáles son sus hipótesis sobre la entrada del algoritmo, pero no es necesario explicar por qué el algoritmo es correcto.

- En caso de utilizar una modificación de un algoritmo visto en la materia, se debe explicar cómo se modifica el algoritmo, cómo cambian (si es que cambian) las hipótesis, y qué problema resuelve esta modificaicón. Mostrar que el algoritmo modificado resuelve el problema que intentaban resolver.

A modo de ejemplo, la siguiente podría ser una lista de las cosas que deberían explicar en un informe.

- El código toma un grafo ponderado como entrada, le agrega las aristas de peso 4 que conectan los vértices que están a distancia exactamente 10, los cuales se calculan utilizando el algoritmo de Dijkstra, y luego se corre una variación del algoritmo de Kruskal que, en lugar de encontrar el árbol generador mínimo, encuentra un bosque generador conformado por exactamente dos árboles y que sea de peso mínimo.
- El código dado resuelve el problema ya que nuestro país es modelado a través de un grafo, donde las ciudades son representadas por los vértices del grafo, las rutas nacionales son representadas por las aristas originales del grafo, y las rutas provinciales, que no son parte de la entrada del problema, son representadas por estas aristas de peso 4 que se le agregan al grafo. El problema que se pide resolver puede ser modelado de esta manera ya que el bosque generador mínimo de exactamente dos árboles estará conformado por árboles que contengan las ciudades donde hará su campaña cada uno de los dos candidatos a director del departamento de Computación, y como el problema pide minimizar la suma de las longitudes de las rutas que sea necesario utilizar para conectar las ciudades de cada candidato, entonces el problema se puede modelar con este algoritmo.
- El algoritmo sólo funciona si los pesos de las aristas son positivos, y siempre que esto pase funciona, pero dado que las aristas tienen como peso longitudes de las rutas, y las rutas no pueden tener, según lo descripto en el enunciado, longitud negativa, entonces el algritmo resuelve correctamente este problema.
- A la hora de resolver este problema se utilizó el algoritmo de Dijkstra, el cual calcula la distancia entre un vértice del grafo y todos los demás. Este algoritmo es utilizado una vez por cada vértice ya que sólo calcula las distancias de un vértice dado a todos los demás. Este algoritmo resuelve correctamente el problema de encontrar distancias, y por lo tanto nos sirve para encontrar los pares de vértices que se encuentran a distancia 10.
- También se utilizó una modificación del algoritmo de Kruskal, el cual calcula árboles generadores mínimos, y en este caso se utilizó para encontrar el bosque generador mínimo de exactamente dos árboles. A este item se le debería agregar una explicación de por qué dicha modificación funciona.

Cada una debe estar bien desarrollada, en este ejemplo se explica todo de manera muy breve, omitiendo detalles que forman parte de la demostración (los cuales no deben estar omitidos en los trabajos prácticos) y basándose en un enunciado inventado (el cuál ni siquiera está escrito en ningún lado) ya que es una guía para que se entienda la consigna.

Tampoco es necesario que dividan el informe en estos ítems, con que estén todos presentes alcanza. No es necesario un nivel estricto de formalidad, pero sí deberá ser riguroso. Si en la demostración de correctitud se omite un caso importante, más allá de que el caso sea resuelto correctamente, se asumirá que no se tuvo en cuenta ese caso y por lo tanto la demostración será considerada como incompleta, lo cual no necesariamente implicará que el ejercicio esté desaprobado pero no se podrá obtener la nota máxima. A modo de ejemplo: Se escribe un código que trabaja con grafos bipartitos pero en ningún lugar del informe se menciona que el grafo sobre el cual se está trabajando es bipartito ni por qué lo es, o se menciona pero no se hace mención a propiedades sobre ciclos. En la demostración de correctitud se utiliza alguna propiedad sobre la longitud de los ciclos que sólo es correcta si dichas longitudes son pares, pero no se menciona este hecho ni se explica que los grafos bipartitos no contienen ciclos impares. En este caso la demostración será considerada incompleta. Ante la duda y si no están seguros, siempre vayan por el camino de la demostración formal, y si les parece que formalizar la demostración puede ser muy complicado, siempre tienen la posibilidad de consultar a los docentes si en ese caso es o no es necesaria la formalidad y qué grado de formalidad.

También se espera del informe un análisis de complejidad, en el que se calcule la complejidad del código, pudiendo utilizar como resultado las complejidades de los algoritmos vistas en clase, sin tener que demostrarlas pero sí mencionándolas, y demostrando la complejidad del código de manera más general. Por

ejemplo, si se utiliza el algoritmo de Dijkstra sobre un grafo que no es explícitamente dado en la entrada, no alcanza con citar la complejidad del algoritmo de Dijkstra sino que también se debe tener en cuenta la complejidad de generar el grafo para el cual se corre el algoritmo en función de la entrada del problema.

En caso de modificar un algoritmo visto en clase, se debe explicar cómo la modificación del algoritmo cambia (o no) la complejidad del mismo.

Al igual que en la parte de correctitud, no se espera demasiada formalidad a la hora de demostrar complejidades pero sí que transmitan que entienden lo que están haciendo. Nuevamente, ante la duda, siempre pueden recurrir a la demostración más formal posible o consultar al respecto.

Por último, se espera que indiquen qué instancias (o familias de instancias) utilizaron para testear su algoritmo. No es necesario que entreguen estas instancias, y si deciden hacerlo deberán entregar un código que las genere y no los archivos de cada instancia.

En caso de utilizar algún flag de compilación específico, o alguna consideración que haya que tener en cuenta a la hora tanto de compilar como de ejecutar el código, deberán entregar un archivo de texto con las instrucciones de cómo compilar y ejecutar el programa desde la terminal de Linux.

Sobre las fechas de entrega, devolución y reentrega

Cada trabajo práctico tendrá una fecha de entrega, de devolución, y dos fechas de reentrega con el siguiente formato (a modo de ejemplo están las fechas del primer trabajo práctico):

Fecha límite de entrega: Viernes 9 de septiembre, hasta las 17:00 hs.
Fecha estimada de devolución: Tres semanas después de la fecha en la que es entregado el TP.
Primer fecha de reentrega: Dos semanas después de devueltas las correcciones, hasta las 17:00 (viernes) o hasta las 22 (martes).
Segunda fecha de reentrega: Viernes 9 de diciembre, hasta las 17:00 hs.

Tanto la fecha límite de entrega como la segunda fecha de reentrega (fecha límite para la última oportunidad de reentrega) serán fechas exactas. La fecha de devolución dependerá de cuándo entreguen, es decir, si entregan un martes se les devolverá el martes y si entregan un viernes se les devolverá un viernes, pudiendo elegir que se les devuelva a la clase siguiente (por ejemplo, si entregaron un martes se les puede devolver el viernes siguiente a la fecha de devolución, y si entregan un viernes se les puede devolver el martes siguiente a la fecha de devolución). La primer fecha de reentrega dependerá de cuándo se les devuelva el TP. Si deciden postergar una clase la devolución (porque no pueden ir el martes o el viernes) deberán avisar, y se moverá la fecha de devolución de su TP. Si pasada la fecha de devolución no van a buscar su TP no se modificará la primer fecha de reentrega.

Tanto la entrega como la reentrega del TP deberán hacerse por correo electrónico (tanto código como informe) a `ltaravilse+pap@gmail.com` y `melaniesclar+pap@gmail.com` antes de la fecha límite correspondiente. El asunto del mail deberá ser Entrega TPN (siendo N el número de TP) o Entrega RTPN, y deberán adjuntar un archivo (`.zip`, `.tar.gz`...) con todos los códigos y el informe del TP. Además deberán entregar el informe impreso.

La entrega del informe podrá hacerse tanto en día martes como en día viernes, siempre en el horario de clases, y pueden entregar el informe el martes (o viernes) y mandar el mail (sin modificar el informe) antes del viernes (o martes) que tengan como fecha límite. Si modifican el código después de la entrega del informe, el código que entreguen deberá ser consistente con el informe. La idea de esto es que si saben que no llegaron con el código, pero no pueden venir el viernes, puedan terminar el informe para el martes, basado en el código que todavía no terminaron, y entregar por mail el viernes (o al revés).

La fecha de devolución es estimada, si llegado el caso llegasen a estar todos los TPs corregidos una semana antes, podría llegar a adelantarse, y si no llegasen a estar para la fecha estimada, se podría llegar a postergar, como máximo, una semana. La primer fecha de reentrega dependerá estrictamente de la fecha real de devolución y no de la estimada.

Sobre los enunciados

El enunciado contará con una breve introducción en la que se mencionen las fechas de entrega, devolución

y reentrega, la cantidad de problemas del TP (que será 3 o 4 dependiendo de cada TP), y los temas a evaluarse en el mismo.

Luego, para cada uno de los problemas el enunciado constará de:

- Enunciado en lenguaje coloquial (una descripción en castellano del cuál deberán extraer una interpretación formal).
- Complejidad esperada: Es criterio de aprobación respetar esta complejidad, es decir, si no se respeta (o si el código la respeta pero esto no lo manifiestan en el informe), el ejercicio no está aprobado.
- Formato de entrada, indicando cómo viene la entrada, qué datos contiene y restricciones (por ejemplo, que los números de la entrada son enteros positivos y no enteros en general).
- Formato de salida, indicando cómo se debe escribir la salida del programa. Tanto la entrada como la salida deberán ser por entrada standard/salida standard y se utilizarán formatos para los cuales ningún lenguaje tenga ventajas sobre otros ni sea complicado manejar la entrada/salida.
- Ejemplos. Tanto la entrada del ejemplo, como la salida del ejemplo y la explicación de por qué la respuesta es correcta.
- Pesos mínimos y máximos de cada ejercicio, explicados en la siguiente sección.
- Cotas recomendadas para testear, ya mencionadas anteriormente.
- En algunos casos puede haber pistas que los orienten a la hora de resolver el problema.

Los enunciados no sufrirán modificaciones de la entrega original a la reentrega, es decir, el RTP tendrá exactamente el mismo enunciado que el TP.

Sobre el método de evaluación

Cada TP se aprobará obteniendo una nota de 5 o más en cada uno de los ejercicios. Una vez aprobados todos los ejercicios del TP, la nota del TP será un promedio de la nota de cada ejercicio, ponderado por los pesos que elijan para cada ejercicio. Estos pesos deberán ser elegidos antes de la primer fecha de entrega (deberán aparecer en el informe) y deberán ser números enteros comprendidos entre el peso mínimo y el peso máximo permitidos por cada enunciado (inclusive en ambos casos). En caso de no elegir pesos, el peso que se utilizará para cada ejercicio será un promedio de los pesos elegidos por todos los grupos que si hayan elegido pesos.

Los ítems a evaluar en cada ejercicio serán:

- Código: No sólo debe ser correcto sino que también debe ser prolijo y fácil de leer y entender y bien modularizado, y en el caso de escribir código que no sea sencillo de entender (por la naturaleza del algoritmo, no porque no lo escribieron de manera prolija) deberán agregar comentarios que lo hagan fácil de entender.
- Demostración: deberá quedar claro que entienden por qué lo que hacen es correcto.
- Análisis de complejidad: Deberá quedar claro que entienden por qué el código cumple con la complejidad que dicen que tiene.

Cada una de estas secciones tiene un criterio mínimo de aprobación y deberán estar las tres aprobadas, además de que la nota deberá ser al menos 5. Cuánto pese cada una de estas secciones y qué cosas se tendrán más en cuenta de cada una dependerán de cada ejercicio y serán las mismas para entrega y reentrega, y si bien no se harán públicas se detallará en la devolución cuáles fueron los criterios bajo los cuales se bajó nota como para que tengan todas las herramientas para mejorarlo en la reentrega.

Si entregan pasada la primer fecha de reentrega, ya no podrán obtener la nota máxima de 10 en el TP, sino que la nota será un promedio ponderado de la nota original del TP (0 si no se entregó) y la nota de la reentrega, con pesos 0.2 y 0.8 respectivamente (es decir $0{,}2 \times N_{TP} + 0{,}8 \times N_{RTP}$ siendo N_{TP} la nota original y N_{RTP} la nota de la reentrega).