



IIC2133 — Estructuras de Datos y Algoritmos

2019 - 1

Tarea 4

Fecha de entrega código: 7 de Julio

Motivación

Para esta tarea quisimos hacer algo distinto, por lo que tendremos un concurso de programación donde los mejores se podrán llevar una salida a tomar brunch a la tetería auspiciada por su equipo docente junto con tazas del capítulo y otros premios. Además todos tendrán la oportunidad de tener una nota máxima 9!



Figura 1: Imagen promocional de la tetería

Temáticas

Esta tarea consiste en 10 problemas de programación cubriendo diversos temas, como programación dinámica, DFS, BFS, Dijkstra, Grafos, Max Flow, Hashing, Conjuntos Disjuntos, Heaps y más. Su nota dependerá de cuantos problemas logren resolver, donde cada problema vale un punto, y pasado el séptimo, valen 0.5 (su nota se calcula como 1 + Puntaje). Un problema solo contará como resuelto cuando el juez lo acepte (más de esto adelante) por lo que no existirán puntajes parciales por un problema.

Plataforma

Para esta tarea hemos implementado un servidor que recibirá su código y lo corregirá automáticamente, esto significa dos cosas:

1. No habrán recorreciones de esta tarea
2. Ustedes mismos podrán saber la nota de su tarea en tiempo real

Para ingresar al servidor deben ingresar a esta dirección: edd.ing.puc.cl/boca/ con su usuario UC y número de alumno como clave. **Deben ingresar a *Options* y cambiar su clave por una más segura por favor.**

Subir una solución

Para subir una solución al servidor, deben ingresar a *Runs*, elegir el problema para el que quieren evaluar su solución y el lenguaje de programación. Después de eso el servidor corregirá automáticamente su código y les entregará la solución. Nota que este proceso puede demorar hasta un par de minutos aunque normalmente es rápido (30 segundos).

Respuestas

Dependiendo de su solución, el servidor les entregará cualquiera de las siguientes respuestas:

1. Not Answered Yet: Esto significa que el servidor aún no corrige su código, solo tienen que ser pacientes
2. YES: Esto significa que su solución es correcta y tienen ese problema bueno
3. NO - Compilation Error: (esto aplica para C/C++) Esto significa que su código no pudo ser compilado
4. NO - Runtime Error: Esto significa que su código se cayó durante la ejecución
5. NO - Time Limit Exceeded: Esto significa que su código es muy lento y no pasó los casos de prueba dentro del tiempo exigido
6. NO - Presentation Error: Esto significa que su solución es correcta pero no el formato de su output es distinto al que esperaba el juez (revisen espacios en blanco, saltos de linea, etc)
7. NO - Contact Staff: Deben enviarle un email a ithermosilla@uc.cl, esto sucede cuando el servidor detecta casos de copia (puede que hayan casos de falsos positivos)

Scoreboard

El servidor mantendrá un ranking en vivo de los alumnos de esta tarea el que también usaremos para asignar los premios. El orden se calcula de la siguiente manera

- Número de problemas resueltos
- En caso de empate, tiempo total de los problemas resueltos donde tiempo es la suma total de los tiempos de sus problemas resueltos. El tiempo de cada problema es el número de minutos que han pasado desde el inicio del torneo hasta que el problema fue aceptado por el servidor. Cada *submision* incorrecta penalizará su tiempo de ese problema en 20 minutos adicionales.

Otros menus del servidor

Los otros menús del servidor pueden ser ignorados por ustedes; *Clarifications, Tasks y Backups*

Premios

Como mencionábamos al inicio, el ganador del concurso se ganará una ida a comer para él y una persona más a la tetería (local de brunch). Además, o todas las personas que logren resolver 10 problemas o los primeros 8 lugares, lo que sea que sea menor, se llevarán como premio una taza del capítulo DCC. Es importante notar que ninguno de los aspectos competitivos de esta tarea afectan en su nota, por lo que si no están interesados, no queremos que estén preocupados al respecto.

Lenguajes, Input y Output

Para esta tarea podrán elegir entre usar C, C++, Java, Python2 y Python3 para su tarea, usen el que más les acomode. Su programa tendrá que leer el input a través del STDIN y entregar su output a través del STDOUT, se adjuntan ejemplos a continuación de esto para varios lenguajes.

Input: 1 2 3

```
// C
int a, b, c;
scanf("%d %d %d", &a, &b, &c); // Leer input del STDIN
printf("%d %d %d\n", a, b, c); // Entregar output al STDOUT
```

```

// C++
#include <bits/stdc++.h>
using namespace std;

cin >> a >> b >> c; // Leer input del STDIN
cout << a << " " << b << " " << c << "\n"; // Entregar output al STDOUT

# Python 3
a, b, c = [int(x) for x in input().split()] # Leer input del STDIN
print("{} {} {}".format(a, b, c)) # Entregar output al STDOUT

```



Figura 2: Imagen promocional de una taza del capítulo

Testear su código antes de subirlo

Naturalmente querrán probar su código antes de subirlo al servidor, la mejor manera de hacer esto es crear un archivo `in.txt` en la misma carpeta que su código, copiar y pegar input de ejemplo y correr el siguiente comando:

```

~ python a.py < in.txt
~ python3 a.py < in.txt
~ gcc a.c & a.out < in.txt
~ g++ a.c & a.out < in.txt

```

Estrategia general para la tarea

Hay 10 problemas con diversas dificultades y temáticas y no es necesario resolverlos en un orden particular, por lo que les recomendamos leer todos los problemas y partir por los que les sean más fáciles, ya sea por que están más familiarizados con el tema o directamente la dificultad del problema es más fácil. Por eso adjuntamos junto a cada problema una estimación de la dificultad de esta/

Problemas

Los enunciados de los problemas pueden ser descargados de la sección *problems*. Mucha suerte en su tarea.