

ICI-142 Fundamentos de Programación

Tarea #5

Wenceslao Palma <wenceslao.palma@ucv.cl>

Una empresa que se encuentra en período de negociación propone a sus trabajadores una nueva forma de calcular los salarios. Esta nueva forma de calcular considera que el salario de un trabajador se calcula considerando el precio que tiene cada tarea y la dependencia que pueda existir entre ciertas tareas.

Una tarea X depende de una tarea Y si: (i) X depende directamente de Y o (ii) si existe una tarea Z tal que X depende directamente de Z y Z depende de Y . Considere que no existe circularidad en la relación de dependencia y que una tarea puede ser realizada por más de un trabajador. Luego, el sueldo de un trabajador será la suma de los precios de las tareas que realiza más el precio de todas las tareas que dependen de las tareas que él realiza.

Para el desarrollo de la tarea considere el uso de las siguientes estructuras de datos:

```
typedef struct trab{
    int numTrabajos; /* cantidad de trabajos que realiza un trabajador*/
    int *tareas; /* almacena las tareas que realiza el trabajador*/
}trabajador;

/*puntero que contiene la @ de memoria de un arreglo dinámico de trabajadores*/
trabajador *trabajadores;

typedef struct tar{
    int precio; /* precio de la tarea t*/
    int numTareasAsociadas; /* cantidad de tareas que dependen de t*/
    int *tareasAsociadas; /* tareas que dependen de t*/
    int numTrabajadores; /* número de trabajadores que realiza t*/
    int *trabajadorAsociado; /* cuales son los trabajadores que realizan t*/
}tarea;

/* puntero que contiene la @ de memoria de un arreglo dinámico de tareas*/
tarea *tareas;
```

Entonces el programa a desarrollar debe calcular el salario para cada trabajador.

Entrada

La entrada se encuentra en un archivo de texto llamado **in.txt** el cual contiene un conjunto de casos de prueba. La primera línea del archivo contiene 2

enteros T y W que indican el número de tareas y empleados ($1 \leq T \leq 100$ y $1 \leq W \leq 100$). Las tareas son numeradas desde 1 hasta T y los trabajadores desde 1 hasta W . Luego, una secuencia de líneas describe las tareas desde la 1 hasta la T en orden ascendente. Cada tarea es descrita usando 2 líneas. La primera esta compuesta de 3 enteros P ($1 \leq P \leq 1000$), NT ($0 \leq NT \leq T$) y NW ($1 \leq NW \leq W$) que representan respectivamente el precio a pagar por una tarea, el número de tareas que dependen de ella y el número de empleados que la realiza. La segunda se compone de $NT + NW$ enteros, los NT primeros corresponden a las tareas que dependen de ella y los NW últimos corresponden a los empleados que desarrollan la tarea. La entrada termina cuando $T = W = 0$.

Salida

La salida del programa se debe almacenar en un archivo de texto llamado **out.txt**. Para cada caso de prueba se debe escribir en el archivo:

- una línea que contenga tres # indicando el comienzo de cada caso.
- para cada trabajador, una línea que contenga dos enteros i y s , separados por un espacio, indicando que el trabajador i recibe s como remuneración.

Ejemplo

Dada la siguiente entrada:

```

3 2
100 2 2
2 3 1 2
40 0 1
1
60 0 1
2
7 2
10 2 1
2 3 1
10 2 1
4 5 2
10 2 1
6 7 2
10 0 1
1
10 0 1
1
10 0 1
1
10 0 1
1
0 0

```

La salida es:

```
###
1 200
2 200
###
1 70
2 60
```

Restricciones

- La tarea debe ser codificada en Lenguaje C. No utilice funciones que no pertenecen al ANSI C.
- Para compilar utilice gcc/mingw.
- El desarrollo de la solución debe considerar el uso de las estructuras de datos propuestas, funciones y paso de parámetros.
- Se debe respetar las especificaciones entregadas para la entrada y salida.
- La corrección incluye una interrogación.
- Solo se recibirán tareas fuera de plazo dentro de las 24 horas siguientes a la fecha de entrega. Nota máxima es un 5.0

Fecha de entrega : Miércoles 31 de Octubre, código fuente **tarea5.c** indicando en su interior como comentario su nombre y rut, enviar por email hasta las 24h00.

email grupo 1 <ici142-1@inf.ucv.cl>

email grupo 2 <ici142-2@inf.ucv.cl>