



C Piscine

Rush 01

Resumen: Este documento corresponde al enunciado del Rush 01 de la C Piscine de 42.

Versión: 7.1

Índice general

I.	Instrucciones	2
II.	Preámbulo	4
III.	Enunciado	5
IV.	Anexo	7
V.	Entrega y evaluación	9

Capítulo I

Instrucciones

- El grupo se registrará automáticamente para la evaluación.
- No canceles la evaluación, no tendrás una segunda.
- Toda petición de precisiones sobre los enunciados complicará los ejercicios.
- Debéis respetar el procedimiento de entrega para todos vuestros ejercicios.
- Este enunciado puede cambiar hasta una hora antes de la entrega.
- La Moulinette compila con los flags -Wall -Wextra -Werror y utiliza gcc.
- Si vuestro programa no compila, tendréis 0.
- Vuestro programa debe estar escrito de acuerdo a la Norma. Si tienes archivos y/o funciones, están incluidos en esta regla y tendréis un 0 si contienen algún error de norma.
- Tienes que realizar este proyecto con resto de miembros del equipo impuesto y debéis presentaros todos a la evaluación a la hora acordada.
- El proyecto debe estar terminado cuando se presente a la evaluación. El propósito de la evaluación es que presentéis y expliquéis vuestro trabajo en detalle.
- Cada miembro del grupo tendrá que estar perfectamente al corriente del trabajo realizado. Si elegís dividir el trabajo, aseguraos de que todos entendéis lo que ha hecho el resto. Durante la evaluación se harán preguntas y la nota del grupo se basará en la explicación peor.
- Evidentemente, es tu responsabilidad reunir al equipo de trabajo. Tienes todos los medios disponibles para contactar con el resto de miembros del equipo: teléfono, e-mail, paloma mensajera, sesión de espiritismo, etc. No se aceptará ninguna excusa en lo que respecta a los problemas de grupo. La vida es injusta, pero es lo que hay.
- De todas maneras, si después de haberlo intentado realmente todo no puedes contactar con un miembro de tu grupo: haced el rush y entregadlo. Intentaremos encontrar una solución durante la evaluación. Incluso si el integrante que falta es el team leader, todos tenéis acceso al repositorio.
- Obviamente, vuestro trabajo deberá respetar La Norma. ¡Sed muy rigurosos!

- ¡Disfrutad!

Capítulo II

Preámbulo

He aquí lo que nos cuenta Wikipedia sobre el perezoso de dos dedos:

El perezoso de dos dedos tiene la reputación de ser el animal más lento del mundo. El adulto pesa de media entre 4 y 8 kg y tiene el tamaño de un perro pequeño: 60 a 85 cm de largo con una cola de 1,4 a 3,3 cm. Presenta un cuello corto, 4 patas largas del mismo tamaño rematadas con 2 garras curvas en las delanteras y tres en las traseras, que pueden llegar a los 7 cm de largo. La cabeza es corta y chata, con una nariz respingona, orejas rudimentarias y ojos grandes.


Se pasan cerca del 80 muy lentamente. Duermen, comen, se aparean, paren y cuidan de las crías sin jamás bajar de su árbol, permaneciendo colgados de las ramas con la cabeza hacia abajo. Rara vez bajan al suelo, solo para cambiar de árbol cuando la comida escasea o para defecar, una vez por semana.

Están perfectamente camuflados en los árboles, su cuerpo pardo verdoso recogido sobre sí mismo se confunde fácilmente con un nido de termitas o con una excrecencia de la madera ante los depredadores. En caso de ataque, se defienden con garras y dientes aunque suelen ser pacíficos y cuentan más con sus capacidades para mimetizarse con el entorno. Se quedan inmóviles durante largas horas pero se rascan con frecuencia, desvelando a veces así su presencia.

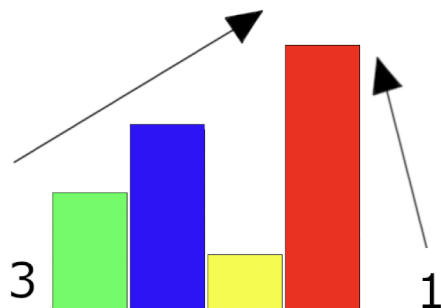
La cultura del cine no ayudará a este proyecto aunque sea importante.

Capítulo III

Enunciado

	Ejercicio: 00
Rush-01	
Directorio de entrega: <i>ex00/</i>	
Archivos a entregar: Todos los archivos necesarios	
Funciones autorizadas: write, malloc, free	

- Vuestro código fuente será compilado por el comando: `gcc -Wall -Wextra -Werror -o rush-01 *.c`
- Vuestro directorio de entrega tendrá que incluir todo los archivos necesario para la compilación de vuestro programa.
- Cread un programa que resuelva el siguiente problema:
- Sobre un mapa de 4 por 4, coloca cajas de entre 1 y 4 de altura tal forma que en cada línea y columna se vea el número correcto de cajas desde cada punto de vista posible (izquierda/ derecha para las líneas y arriba/abajo para columnas)
- Ejemplo: la caja de tamaño 3 aquí esconde a la caja de altura 1, lo que implica que desde la izquierda solo hay 3 cajas visibles. Desde la derecha, la caja de altura 4 esconde las demás cajas, por lo tanto solo hay una caja visible.



- Cada vista (dos por línea y dos por columna) tendrá un valor determinado, que irá del 1 al 4. Vuestro programa debe colocar las cajas correctamente, cuidando además de no tener más de una caja de cada altura sobre cada línea y columna.
- Vuestra propuesta debe contener la primera solución que encontréis.
- Se ejecutará el programa de la forma siguiente:

```
> ./rush-01 "col1up col2up col3up col4up col1down col2down col3down col4down row1left row2left  
row3left row4left row1right row2right row3right row4right"
```

- (cf. anexo 1)
- col1up corresponde al valor para el punto de vista desde lo alto de la columna de la izquierda. Cada uno de ellos respresenta una cadena de caracteres de valores con un rango entre 1 y 4.
- Esta es la ÚNICA entrada aceptable para vuestro programa. Cualquier otra entrada debe considerarse un error.
- He aquí un ejemplo de entrada/salida prevista para un conjunto válido:

```
> ./rush-01 "4 3 2 1 1 2 2 2 4 3 2 1 1 2 2 2" | cat -e  
1 2 3 4$  
2 3 4 1$  
3 4 1 2$  
4 1 2 3$
```

- (cf. anexo 2 y 3)
- En caso de error, o si no podéis encontrar ninguna solución, escribidá “Error” seguido de un salto de línea.
- Si quieres puntos extra, puedes intentar manejar otro tamaño de mapa (hasta 9x9 !!!!).
- Como es habitual, si un bonus funciona, pero el ejercicio obligatorio no pasa las pruebas, tendréis un 0.

Capítulo IV

Anexo

Lo que sigue es una representación artística de vuestro programa. Por supuesto, tenéis que entregar un programa como el descrito en el capítulo anterior. Estos ejemplos tienen como único objetivo ayudarle a comprender los enunciados.

- Anexo 1:

	col1up	col2up	col3up	col4up	
row1left					row1right
row2left					row2right
row3left					row3right
row4left					row4right
	col1down	col2down	col3down	col4down	

- Representación de la ejecución del programa con col_up, col_down, row_left y row_right
- Anexo 2:

	4	3	2	1	
4					1
3					2
2					2
1					2
	1	2	2	2	

- Remplazando los col* y los row*, se obtiene esto.
- Anexo 3:

	4	3	2	1	
4	1	2	3	4	1
3	2	3	4	1	2
2	3	4	1	2	2
1	4	1	2	3	2
	1	2	2	2	

- Su programa debe rellenar las casillas internas y devolver la respuesta tal y como se pide en los enunciados.

Capítulo V

Entrega y evaluación

Entrega tu proyecto en tu repositorio `Git` como de costumbre. Solo el trabajo entregado en el repositorio será evaluado durante la defensa. No dudes en comprobar varias veces los nombres de los archivos para verificar que sean correctos.

Como este proyecto no es comprobado por un programa, puedes organizar tus archivos como consideres oportuno, siempre y cuando entreges los archivos obligatorios y estos cumplan con los requisitos.



Sólo necesitas entregar los archivos requeridos por el enunciado de este proyecto.