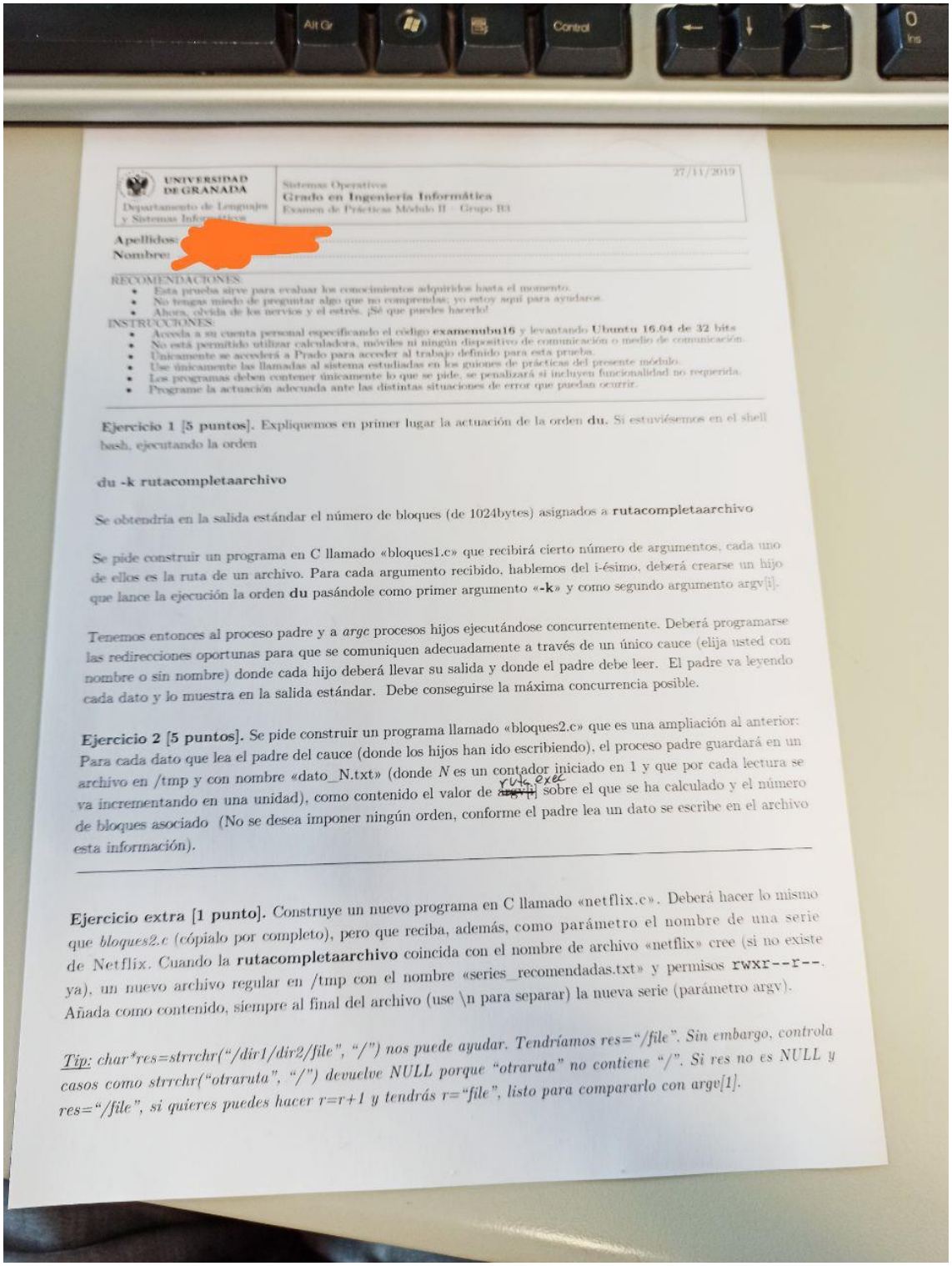

UNIVERSIDAD DE GRANADA
Departamento de Lenguajes y Sistemas Informáticos

Sistemas Operativos
Grado en Ingeniería Informática
Examen de Prácticas Módulo II – Grupo B3

27/11/2019

Apellidos: [redacted]
Nombre:

RECOMENDACIONES:
- Esta prueba sirve para evaluar los conocimientos adquiridos hasta el momento.
- No tengas miedo de preguntar algo que no comprendas; yo estoy aquí para ayudarte.
- Ahora, olvida de los nervios y el estrés. ¡Sé que puedes hacerlo!

INSTRUCCIONES:
- Acceda a su cuenta personal especificando el código **examenubu16** y levantando **Ubuntu 16.04 de 32 bits**
- No está permitido utilizar calculadora, móviles ni ningún dispositivo de comunicación o medio de comunicación.
- Únicamente se accederá a Prado para acceder al trabajo definido para esta prueba.
- Use únicamente las llamadas al sistema estudiadas en los guiones de prácticas del presente módulo.
- Los programas deben contener únicamente lo que se pide, se penalizará si incluyen funcionalidad no requerida.
- Programe la actuación adecuada ante las distintas situaciones de error que puedan ocurrir.

Ejercicio 1 [5 puntos]. Expliquemos en primer lugar la actuación de la orden **du**. Si estuviésemos en el shell bash, ejecutando la orden

du -k rutacompletaarchivo

Se obtendría en la salida estándar el número de bloques (de 1024bytes) asignados a **rutacompletaarchivo**

Se pide construir un programa en C llamado «bloques1.c» que recibirá cierto número de argumentos, cada uno de ellos es la ruta de un archivo. Para cada argumento recibido, hablemos del i-ésimo, deberá crearse un hijo que lance la ejecución la orden **du** pasándole como primer argumento «**-k**» y como segundo argumento argv[i].

Tenemos entonces al proceso padre y a *argc* procesos hijos ejecutándose concurrentemente. Deberá programarse las redirecciones oportunas para que se comuniquen adecuadamente a través de un único cauce (elija usted con nombre o sin nombre) donde cada hijo deberá llevar su salida y donde el padre debe leer. El padre va leyendo cada dato y lo muestra en la salida estándar. Debe conseguirse la máxima concurrencia posible.

Ejercicio 2 [5 puntos]. Se pide construir un programa llamado «bloques2.c» que es una ampliación al anterior: Para cada dato que lea el padre del cauce (donde los hijos han ido escribiendo), el proceso padre guardará en un archivo en /tmp y con nombre «dato_N.txt» (donde *N* es un contador iniciado en 1 y que por cada lectura se va incrementando en una unidad), como contenido el valor de ~~argv[i]~~ ruta.exe sobre el que se ha calculado y el número de bloques asociado (No se desea imponer ningún orden, conforme el padre lea un dato se escribe en el archivo esta información).

Ejercicio extra [1 punto]. Construye un nuevo programa en C llamado «netflix.c». Deberá hacer lo mismo que *bloques2.c* (cópialo por completo), pero que reciba, además, como parámetro el nombre de una serie de Netflix. Cuando la **rutacompletaarchivo** coincida con el nombre de archivo «netflix» cree (si no existe ya), un nuevo archivo regular en /tmp con el nombre «series_recomendadas.txt» y permisos `rwxr--r--`. Añada como contenido, siempre al final del archivo (use \n para separar) la nueva serie (parámetro argv).

*Tip: char*res=strrchr("/dir1/dir2/file", "/") nos puede ayudar. Tendríamos res="/file". Sin embargo, controla casos como strrchr("otraruta", "/") devuelve NULL porque "otraruta" no contiene "/". Si res no es NULL y res="/file", si quieres puedes hacer r=r+1 y tendrás r="file", listo para compararlo con argv[1].*