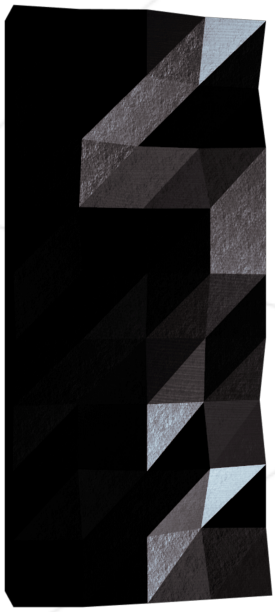

ft_printf

Porque ft_putnbr() y ft_putstr() no son suficientes

Resumen:
El objetivo de este proyecto es bastante sencillo. Deberás reimplementar printf(). Aprenderás, principalmente, a utilizar un número variable de argumentos. ¿Mola, verdad? ¡Pues sí que mola! :)

Versión: 12.0

Índice general

Capítulo I

Introducción

Vas a descubrir una función de `C` muy famosa y versátil: `printf()`. Este proyecto es una gran oportunidad para mejorar tus habilidades de programación. Es un proyecto de dificultad moderada.

También descubrirás las **funciones variádicas** en `C`.

La clave para superar `ft_printf` es tener un código bien estructurado y extensible.

`Una vez que superes este proyecto, podrás incluir ft_printf() en tu libft, por lo que podrás utilizarla en futuros proyectos en C.`

Capítulo II

Instrucciones generales

- Tu proyecto deberá estar escrito en C.
- Tu proyecto debe estar escrito siguiendo la Norma. Si tienes archivos o funciones adicionales, estas están incluidas en la verificación de la Norma y tendrás un `0` si hay algún error de norma en cualquiera de ellos.
- Tus funciones no deben terminar de forma inesperada (segfault, bus error, double free, etc) excepto en el caso de comportamientos indefinidos. Si esto sucede, tu proyecto será considerado no funcional y recibirás un `0` durante la evaluación.
- Toda la memoria asignada en el heap deberá liberarse adecuadamente cuando sea necesario. No se permitirán leaks de memoria.
- Si el enunciado lo requiere, deberás entregar un `Makefile` que compilará tus archivos fuente al output requerido con las flags `-Wall`, `-Werror` y `-Wextra`, utilizar cc y por supuesto tu `Makefile` no debe hacer relink.
- Tu `Makefile` debe contener al menos las normas `$(NAME)`, `all`, `clean`, `fclean` y `re`.
- Para entregar los bonus de tu proyecto deberás incluir una regla `bonus` en tu `Makefile`, en la que añadirás todos los headers, librerías o funciones que estén prohibidas en la parte principal del proyecto. Los bonus deben estar en archivos distintos `_bonus.{c/h}`. La parte obligatoria y los bonus se evalúan por separado.
- Si tu proyecto permite el uso de la `libft`, deberás copiar su fuente y sus `Makefile` asociados en un directorio `libft` con su correspondiente `Makefile`. El `Makefile` de tu proyecto debe compilar primero la librería utilizando su `Makefile`, y después compilar el proyecto.
- Te recomendamos crear programas de prueba para tu proyecto, aunque este trabajo **no será entregado ni evaluado**. Te dará la oportunidad de verificar que tu programa funciona correctamente durante tu evaluación y la de otros compañeros. Y sí, tienes permitido utilizar estas pruebas durante tu evaluación o la de otros compañeros.
- Entrega tu trabajo en tu repositorio `Git` asignado. Solo el trabajo de tu repositorio `Git` será evaluado. Si Deepthought evalúa tu trabajo, lo hará después de tus com-

pañeros. Si se encuentra un error durante la evaluación de Deepthought, esta habrá terminado.

Capítulo III

Instrucciones sobre la IA

Contexto

Este proyecto está diseñado para ayudarte a descubrir los fundamentos que construirán tu formación en TIC (lo que conocemos como Tecnologías de la Información y la Comunicación).

Para afianzar los conocimientos y habilidades clave, es esencial adoptar un enfoque reflexivo sobre el uso de herramientas de IA.

El auténtico aprendizaje de los fundamentos requiere un esfuerzo intelectual real, a través de desafíos, repetición y el intercambio de conocimiento que procede del aprendizaje entre pares.

Para una visión más completa de nuestra postura sobre la IA (ya sea como herramienta de aprendizaje, como parte del plan de estudios de TIC o como una expectativa en el mercado laboral) puedes consultar las preguntas frecuentes dedicadas en la intranet.

Mensaje principal:

- Construir fundamentos sólidos sin atajos.
- Desarollar de forma real habilidades técnicas y transversales.
- Experimentar el aprendizaje entre pares de forma inmersiva, empezando por aprender a aprender y por resolver nuevos problemas.
- El proceso de aprendizaje es más importante que el resultado.
- Aprender sobre los riesgos asociados a la IA y desarrollar prácticas de control efectivas y medidas que neutralicen los errores comunes.

Reglas para estudiantes:

- Aplica la lógica y el razonamiento a las tareas asignadas, especialmente antes de recurrir a la IA.
- No deberías pedir respuestas directas a la IA.
- Infórmate sobre el enfoque global de 42 respecto la IA.

Resultados de esta etapa:

Durante esta fase de construcción de los fundamentos, conseguirás:

- Obtener una base adecuada en tecnología y en programación.
- Comprender por qué y cómo la IA puede ser peligrosa durante esta fase.

Comentarios y ejemplos:

- Sí, sabemos que la IA existe.Y si, también sabemos que puede resolver tus proyectos. Pero estás aquí para aprender, no para demostrar que la IA ha aprendido. No pierdas tiempo solo para demostrar que la IA puede resolver un problema determinado.
- Aprender en 42 no tiene nada que ver con saber una respuesta, sino con desarrollar las habilidades para encontrarla. La IA te dará la respuesta directa, lo que impide que desarrolles tu propio razonamiento. Razonar requiere tiempo, esfuerzo y cometer errores. Nadie dijo que el proceso iba a sér fácil.
- Ten en cuenta que, durante los exámenes, no tendrás acceso a la IA (no tenemos internet ni dispositivos inteligentes). Te vas a dar cuenta rápidamente si has confiado demasiado en la IA durante tu proceso de aprendizaje de la forma más directa: frente a una hoja en blanco donde vas a tener que escribir tu propio código.
- El aprendizaje entre pares te expone a diferentes ideas y enfoques, mejorando tus habilidades transversales y tu capacidad de pensar de forma diferente. Eso es mucho más valioso que sentarte a chatear con un bot. Así que, ¡sin miedo! Habla, haz preguntas y aprende con el resto de estudiantes.
- Sí, la IA formará parte del plan de estudios, tanto como herramienta de aprendizaje como tema en sí mismo. Incluso tendrás la oportunidad de crear tu propio software de IA. Para aprender más sobre nuestro enfoque progresivo, puedes consultar la documentación disponible en la intranet.

✓ Buenas prácticas:

Me atasco en un nuevo concepto. Le pregunto a alguien cercano cómo lo ha abordado. Hablamos durante 10 minutos y, de repente, todo encaja. Lo entiendo. No entiendo algo concreto del proyecto y no sé cómo continuar. Le pregunto a otra persona cómo lo ha abordado, hablamos sobre el tema y, si es necesario, incluso utilizamos otros métodos (papel y boli, dibujos, metáforas, etc.) hasta conseguir entenderlo.

✗ Mala práctica:

Utilizo la IA en secreto, copio un código que parece correcto. Durante la evaluación entre pares, no puedo explicar nada. Suspendo. Durante el examen, sin IA, me vuelvo a atascar. Suspendo.

Capítulo IV

Parte obligatoria

Nombre de programa	`libftprintf.a`
Archivos a entregar	`Makefile, *.h, */*.h, *.c, */*.c`
Makefile	`NAME, all, clean, fclean, re`
Funciones autorizadas	`malloc, free, write,` `va_start, va_arg, va_copy, va_end`
Se permite usar libft	`Yes`
Descripción	`Escribe una librería que contenga la función` `ft_printf(), que imite el printf() original`

Se debe reprogramar la función `printf()` de la `libc`.

El prototipo de `ft_printf()` es:

```
int     ft_printf(char const *, ...);
```

Estos son los requisitos:

- No se debe implementar la gestión del 'buffer' del `printf()` original.
- Se deben implementar las siguientes conversiones: `cspdiuxX%`
- La función se comparará con el `printf()` original para verificar su comportamiento.
- Hay que usar el comando `ar` para crear la librería.
  El uso de `libtool` está prohibido.
- El archivo `libftprintf.a` deberá ser creado en la raíz de tu repositorio.

Se deben implementar las siguientes conversiones:

- `%c` para imprimir un solo carácter.
- `%s` para imprimir una cadena de caracteres (como se define por defecto en `C`).
- `%p` el puntero `void *` dado como argumento se imprime en formato hexadecimal.
- `%d` para imprimir un número decimal (base 10).
- `%i` para imprimir un entero en base 10.
- `%u` para imprimir un número decimal (base 10) sin signo.
- `%x` para imprimir un número hexadecimal (base 16) en minúsculas.
- `%X` para imprimir un número hexadecimal (base 16) en mayúsculas.
- `%%` para imprimir el símbolo del porcentaje.

Capítulo V

Requisitos del Readme

Debe incluirse un archivo `README.md` en la raíz del repositorio Git. Su propósito es permitir que cualquier persona que no esté familiarizada con el proyecto (pares, personal, responsables de selección, etc.) pueda entender rápidamente de qué trata el proyecto, cómo ejecutarlo y dónde encontrar más información sobre el tema.
El `README.md` debe incluir, como mínimo:

- La primera línea debe estar en cursiva y decir: *Este proyecto ha sido creado como parte del currículo de 42 por <login1>[, <login2>[, <login3>[...]]].*
- Una sección de "**Descripción**"que presente claramente el proyecto, incluyendo su objetivo y una breve visión general.
- Una sección de "**Instrucciones**"que contenga cualquier información relevante sobre compilación, instalación y/o ejecución.
- Una sección de "**Recursos**"que enumere referencias clásicas relacionadas con el tema (documentación, artículos, tutoriales, etc.), así como una descripción del uso de IA, especificando para qué tareas y en qué partes del proyecto se ha utilizado.

➠ **Podrían requerirse secciones adicionales dependiendo del proyecto** (por ejemplo, ejemplos de uso, lista de características, decisiones técnicas, etc.).

Cualquier contenido extra requerida se listará explícitamente a continuación.

- También debe incluirse una explicación y justificación detalladas sobre la elección del algoritmo y la estructura de datos.

`La elección del idioma queda a tu criterio. Se recomienda escribir en inglés, pero no es obligatorio.`

Capítulo VI

Parte bonus

No es necesario hacer todos los bonus.

Lista de bonus:

- Gestiona cualquier combinación de las siguientes flags: '`-0.`' y el ancho mínimo (field minimum width) bajo todas las conversiones posibles.
- Gestiona las siguientes flags: '`# +`' (sí, uno de ellos es un espacio).

Si quieres completar la parte bonus, piensa en la implementación de las características extras desde el principio. De esta forma, evitarás los peligros de un enfoque ingenuo.

La parte bonus solo podrá evaluarse si la parte obligatoria está PERFECTA. Perfecta significa que la parte obligatoria funciona al completo, sin ningún error ni comportamiento inesperado. Si no has superado todos los requisitos de la parte obligatoria, no se evaluará nada de la parte bonus.

Capítulo VII

Entrega y evaluación

Entrega el proyecto en tu repositorio `Git` como de costumbre. Solo será evaluado el contenido de dentro de tu repositorio `Git`. Se recomienda comprobar dos veces los nombres de los archivos para que sean correctos.

Una vez que este proyecto ses superado, se podrá incluir `ft_printf()` en la `libft`, por lo que se podrá utilizarla en futuros proyectos en `C`.

Durante la evaluación, es posible que se solicite una breve **modificación del proyecto**. Esto puede consistir en ajustar ligeramente el comportamiento, modificar unas cuantas líneas de código o incorporar una característica fácil de implementar.

Puede que este paso **no sea necesario en todos los proyectos**, pero debes tenerlo en cuenta si asi se especifica en la hoja de evaluación.

Este paso sirve para a verificar tu comprensión real de una parte específica del proyecto. La modificación se puede realizar en cualquier entorno de desarrollo que elijas (por ejemplo, tu configuración habitual), y debería ser factible en unos pocos minutos, a menos que se defina un plazo específico como parte de la evaluación.
Por ejemplo, se te puede pedir hacer una pequeña actualización en una función o script, modificar lo que se vería en pantalla o ajustar una estructura de datos para almacenar nueva información, etc.

Los detalles (alcance, objetivo, etc.) se especificarán cada **hoja de evaluación** y pueden variar de una evaluación a otra para el mismo proyecto.

```
++++++++++[>+>+++>+++++++>++++++++++<<<-]>>.>---.+++++++++++.++.+++
+++.--.<++.>------.------------.++++++++++++.<.>++++++.------------
.-------. ++++++++++++++++++.<.>---------------.+++++.+++++++++.---
----------.--.+ ++++++++++++++.--------.++++++++++++.<.>----------
------------.+++.+++ ++++.---.----.++++++++++++++++.--------------
--.-.<.>+++++.+++++.<.>-------...
```