Francisco Rubio Illán

frubio-i

frubio-i@student.42barcelona.com

PRINTF

Este proyecto consiste en desarrollar una **función personalizada**, **ft_printf()**, que **imita** el comportamiento de la función **printf()** de la **biblioteca** estándar de **C**. La función ft_printf() es una parte integral de la librería LIBFT, diseñada para proporcionar una serie de funciones de utilidad que facilitan el desarrollo de programas en C. Al igual que las otras funciones en LIBFT,**ft_printf()** está diseñada para ser **modular** y fácilmente **integrable** en **proyectos** de **C**, ofreciendo una alternativa robusta y extensible a la función **printf()** original.

CARACTERÍSTICAS PRINCIPALES

La reprogramación de **printf()** implica el manejo de un número variable de argumentos y la implementación de varias conversiones de **formato**, como **caracteres**, **cadenas**, **enteros**, números **hexadecimales**, **punteros** y el carácter de **porcentaje**. La función **ft_printf()** debe ser capaz de manejar estos formatos de manera eficiente y precisa, replicando el comportamiento esperado de **printf()** en las condiciones dadas en el subject.

MANIPULACIÓN DE FORMATO

- **%c**: Imprime un solo carácter.
- **%s**: Imprime una cadena de caracteres.
- **%p**: Imprime el valor de un puntero en formato hexadecimal.
- **%d** e **%i**: Imprimen un número entero en base 10.
- **%u**: Imprime un número entero sin signo en base 10.
- **%x** y **%X**: Imprimen un número hexadecimal en minúsculas y mayúsculas, respectivamente.
- **%%**: Imprime el carácter de porcentaje.

FUNCIONES CLAVE

El proyecto se estructura alrededor de varias **funciones auxiliares** que trabajan en conjunto para proporcionar la funcionalidad pedida en el subject del proyecto **ft_printf()**:

ft_check_type(): Esta función es responsable de determinar el **tipo de dato** a imprimir basándose en el carácter de formato proporcionado y llama a la función apropiada para manejar dicho tipo.

Funciones de Impresión

ft_put_chr(): Imprime un único **carácter** en la salida estándar.

ft_put_str(): Imprime una **cadena** completa en la salida estándar, con manejo especial para cadenas nulas.

ft_put_p(): Imprime el **valor** de un **puntero** en formato **hexadecimal**, precedido por **"0x"**, incluyendo el manejo de punteros nulos.

ft_put_di(): Imprime un n**úmero entero decimal**, con soporte para números negativos y el valor mínimo de int.

ft_put_u(): Imprime un **número entero sin signo** en decimal.

ft_put_hex(): Imprime un **número** en formato **hexadecimal**, permitiendo tanto minúsculas como mayúsculas según el especificador de formato.

Consideraciones Adicionales

La implementación de **ft_printf()** permite un manejo efectivo de **argumentos variádicos** en C y una interpretación precisa de los especificadores de formato. Es recomendable igualmente explorar el soporte para flags adicionales, anchos de campo y precisión, alineando aún más estrechamente la función personalizada con las capacidades de la función **printf()** original.

Es una función un poco más compleja de la biblioteca estándar de C, que también sirve como una excelente oportunidad para profundizar en el entendimiento de las **funciones variádicas**, el **manejo de memoria dinámica** y el diseño de código **modular** y extensible en C.

Funciones usadas en el proyecto:

Funciones autorizadas:

- malloc
- free
- write
- va_start
- va_arg
- va_copy
- va_end

ft_printf

Prototipo:

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

Parámetros:

pf_str: Cadena de formato que especifica cómo se deben imprimir los argumentos adicionales.

...: Lista variable de argumentos que corresponden a los especificadores en pf_str.

Valor devuelto:

Número total de caracteres impresos.

Descripción:

Función principal que **procesa** la cadena de **formato y** los **argumentos variables**, llamando a las funciones apropiadas para manejar cada tipo de dato **especificado** por los carácteres de formato encontrados en **pf_str**.

ft_check_type

Prototipo:

```
void ft_check_type(const char *pf_str, int *count, va_list
pf_args);
```

Parámetros:

pf_str: Puntero a la posición actual dentro de la cadena de formato que contiene el carácter de formato específico.

count: Puntero al contador general de caracteres impresos.

pf_args: Lista de argumentos variables pasados a ft_printf.

Valor devuelto:

Nada. Void.

Descripción:

Determina el tipo de dato a imprimir basándose en el carácter de formato proporcionado y llama a la función apropiada para manejar dicho tipo.

ft_put_chr

Prototipo:

int ft_putchar(char c, int *count);

Parámetros:

c: El carácter a enviar.

***count**: Contador general de carácteres impresos.

Valor devuelto:

*count.

Descripción:

Imprime el caracter ’c’ en patalla y actualiza el contador de count (+1 por character).

ft_put_str

Prototipo:

```
void ft_put_str(char *s, int *count);
```

Parámetros:

s: Puntero a la cadena de caracteres a imprimir.

count: Puntero al contador general de caracteres impresos.

Valor devuelto:

Nada. Void.

Descripción:

Imprime la cadena **s** en pantalla y **actualiza** el contador **count** con la cantidad de caracteres impresos. Si **s** es **NULL**, imprime **"(null)"**.

ft_put_p

Prototipo:

```
void ft_put_p(unsigned long ptr, int *count, int onetime);
```

Parámetros:

ptr: Valor del puntero a imprimir.

count: Puntero al contador general de caracteres impresos.

onetime: Variable auxiliar para controlar la impresión inicial de "0x".

Valor devuelto:

Nada. Void.

Descripción:

Imprime el **valor** del **puntero** ptr en formato **hexadecimal**, precedido por **"0x"**. Si **ptr** es **NULL**, imprime **"(nil)"**.

ft_put_di

Prototipo:

```
void ft_put_di(int x, int *count);
```

Parámetros:

x: Entero decimal a imprimir.

count: Puntero al contador general de caracteres impresos.

Valor devuelto:

Nada. Void.

Descripción:

Imprime el **entero** x en **decimal**. **Maneja** números **negativos** imprimiendo un signo menos antes del valor absoluto.

ft_put_u

Prototipo:

```
void ft_put_u(unsigned int x, int *count);
```

Parámetros:

x: Entero sin signo a imprimir.

count: Puntero al contador general de caracteres impresos.

Valor devuelto:

Nada. Void.

Descripción:

Imprime el **entero** sin signo x en decimal.

ft_put_hex

Prototipo:

```
void ft_put_hex(unsigned int x, const char *type, int *count);
```

Parámetros:

x: Número hexadecimal a imprimir.

type: Especifica si se debe imprimir en minúsculas (**'x'**) o mayúsculas (**'X'**).

count: Puntero al contador general de caracteres impresos.

Valor devuelto:

Nada. Void.

Descripción:

Imprime el **número** x en **formato hexadecimal**, utilizando letras **minúsculas** o **mayúsculas** según lo especificado por **type**.