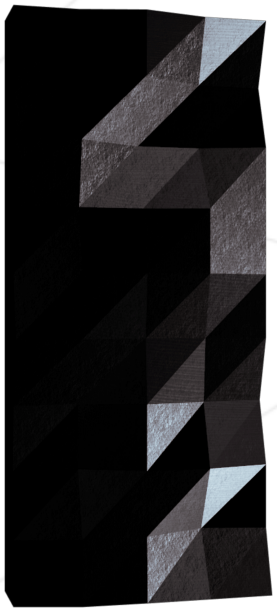

ft_printf

Porque ft_putnbr() e ft_putstr() não são suficientes

Preâmbulo: A meta deste projeto é bastante direta: você irá reimplementar a função printf(). Isso irá principalmente te ensinar como lidar com um número variável de argumentos. Que legal, não é? Na verdade, é bem legal! :)

Versão: 11.0

Sumário

Capítulo I

Introdução

Você explorará uma das funções mais populares e versáteis em `C`: `printf()`. Este exercício proporciona uma excelente oportunidade para aprimorar suas habilidades de programação. Ele é considerado de dificuldade moderada.

Você descobrirá **funções variádicas** em `C`.

A chave para um `ft_printf` bem-sucedido é um código bem estruturado e extensível.

`Assim que você completar com sucesso esta tarefa, você terá permissão para adicionar sua ft_printf() à sua libft, tornando-a disponível para uso em seus projetos em C.`

Capítulo II

Instruções Comuns

- Seu projeto deve ser escrito em C.
- Seu projeto deve ser escrito de acordo com a Norma. Se você tiver arquivos/funções bônus, eles serão incluídos na verificação da norma, e você receberá uma nota `0` se houver algum erro de norma.
- Suas funções não devem sair inesperadamente (segmentation fault, bus error, double free, etc.) exceto por comportamento indefinido. Se isso ocorrer, seu projeto será considerado não funcional e receberá uma nota `0` durante a avaliação.
- Toda memória alocada na heap deve ser liberada corretamente quando necessário. Vazamentos de memória não serão tolerados.
- Se o enunciado exigir, você deve submeter um `Makefile` que compile seus arquivos fonte para a saída requerida com as flags `-Wall`, `-Wextra`, e `-Werror`, usando `cc`. Adicionalmente, seu `Makefile` não deve realizar re-links desnecessários.
- Seu `Makefile` deve conter pelo menos as regras `$(NAME)`, `all`, `clean`, `fclean` e `re`.
- Para submeter bônus para seu projeto, você deve incluir uma regra `bonus` em seu `Makefile`, que adicionará todos os diversos headers, bibliotecas, ou funções que não são permitidas na parte principal do projeto. Os bônus devem ser colocados em arquivos `_bonus.{c/h}`, a menos que o enunciado especifique o contrário. A avaliação das partes obrigatórias e bônus é conduzida separadamente.
- Se seu projeto permitir que você use sua `libft`, você deve copiar suas fontes e seu `Makefile` associado para uma pasta `libft`. O `Makefile` do seu projeto deve compilar a biblioteca usando seu `Makefile`, e então compilar o projeto.
- Nós encorajamos você a criar programas de teste para seu projeto, mesmo que este trabalho **não precise ser submetido e não será avaliado**. Isso lhe dará a oportunidade de testar facilmente seu trabalho e o trabalho de seus colegas. Você achará esses testes especialmente úteis durante sua defesa. De fato, durante a defesa, você está livre para usar seus testes e/ou os testes do colega que você está avaliando.

- Submeta seu trabalho para o repositório Git designado. Somente o trabalho no repositório Git será avaliado. Se o Deepthought for designado para avaliar seu trabalho, isso ocorrerá após as avaliações de seus colegas. Se um erro acontecer em qualquer seção do seu trabalho durante a avaliação do Deepthought, a avaliação será interrompida.

Capítulo III

Instruções de IA

Contexto

Este projeto foi desenvolvido para ajudá-lo a descobrir os blocos de construção fundamentais do seu treinamento em TIC.

Para ancorar adequadamente os conhecimentos e habilidades-chave, é essencial adotar uma abordagem criteriosa ao uso de ferramentas e suporte de IA.

A verdadeira aprendizagem fundamental exige um esforço intelectual genuíno — através de desafios, repetição e trocas de aprendizagem entre pares.

Para uma visão geral mais completa de nossa posição sobre a IA — como ferramenta de aprendizagem, como parte do currículo de TIC e como expectativa no mercado de trabalho — consulte as perguntas frequentes dedicadas na intranet.

Mensagem principal

- Construa bases sólidas sem atalhos.
- Desenvolva verdadeiramente habilidades técnicas e de poder.
- Experimente a verdadeira aprendizagem entre pares, comece a aprender como aprender e resolver novos problemas.
- A jornada de aprendizagem é mais importante que o resultado.
- Aprenda sobre os riscos associados à IA e desenvolva práticas eficazes de controle e contramedidas para evitar armadilhas comuns.

Regras para o aluno:

- Você deve aplicar o raciocínio às suas tarefas atribuídas, especialmente antes de recorrer à IA.
- Você não deve pedir respostas diretas à IA.
- Você deve aprender sobre a abordagem global da 42 em relação à IA.

Resultados da fase:

Nesta fase fundamental, você obterá os seguintes resultados:

- Obter bases sólidas em tecnologia e codificação.
- Saber por que e como a IA pode ser perigosa durante esta fase.

Comentários e exemplo:

- Sim, sabemos que a IA existe — e sim, ela pode resolver seus projetos. Mas você está aqui para aprender, não para provar que a IA aprendeu. Não perca seu tempo (nem o nosso) apenas para demonstrar que a IA pode resolver o problema dado.
- Aprender na 42 não é sobre saber a resposta — é sobre desenvolver a capacidade de encontrar uma. A IA lhe dá a resposta diretamente, mas isso o impede de construir seu próprio raciocínio. E o raciocínio leva tempo, esforço e envolve falhas. O caminho para o sucesso não deve ser fácil.
- Lembre-se de que durante os exames, a IA não estará disponível — sem internet, sem smartphones, etc. Você perceberá rapidamente se confiou demais na IA em seu processo de aprendizagem.
- A aprendizagem entre pares o expõe a diferentes ideias e abordagens, melhorando suas habilidades interpessoais e sua capacidade de pensar de forma divergente. Isso é muito mais valioso do que apenas conversar com um bot. Então não seja tímido — converse, faça perguntas e aprenda juntos!
- Sim, a IA fará parte do currículo — tanto como ferramenta de aprendizagem quanto como um tópico em si. Você até terá a chance de construir seu próprio software de IA. Para saber mais sobre nossa abordagem crescente, consulte a documentação disponível na intranet.

✓ Boa prática:

> Estou travado em um novo conceito. Pergunto a alguém próximo como ele abordou isso. Conversamos por 10 minutos — e de repente, clica. Entendi.

✗ Má prática:

> Uso secretamente a IA, copio algum código que parece certo. Durante a avaliação entre pares, não consigo explicar nada. Eu falho. Durante o exame — sem IA — estou travado novamente. Eu falho.

Capítulo IV

Parte Obrigatória

Nome do programa	`libftprintf.a`
Arquivos para entregar	`Makefile, *.h, */*.h, *.c, */*.c`
Makefile	`NAME, all, clean, fclean, re`
Funções externas autorizadas	`malloc, free, write,` `va_start, va_arg, va_copy, va_end`
Libft autorizada	`Sim`

Descrição	`Escreva uma biblioteca que contenha ft_printf(),`
	`uma função que irá imitar a função printf()`
	`original.`

Você terá que reescrever a função `printf()` da `libc`.

O protótipo de `ft_printf()` é:

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

Aqui estão os requisitos:

- Não implemente o gerenciamento de buffer da `printf()` original.
- Sua função tem que lidar com as seguintes conversões: `cspdiuxX%`
- Sua implementação será avaliada em comparação ao comportamento da `printf()` original.
- Você deve usar o comando `ar` para criar sua biblioteca.
  O uso do comando `libtool` é estritamente proibido.
- `libftprintf.a` deve ser criado na raiz do seu repositório.

Você tem que implementar as seguintes conversões:

- `%c` Imprime um único caractere.
- `%s` Imprime uma string (como definido pela convenção comum de `C`).
- `%p` O argumento ponteiro `void *` tem que ser impresso no formato hexadecimal.
- `%d` Imprime um número decimal (base 10).
- `%i` Imprime um inteiro na base 10.
- `%u` Imprime um número decimal sem sinal (base 10).
- `%x` Imprime um número em hexadecimal (base 16) em minúsculas.
- `%X` Imprime um número em hexadecimal (base 16) em maiúsculas.
- `%%` Imprime um sinal de porcentagem.

Capítulo V

Parte Bônus

Você não precisa fazer todos os bônus.

Lista de bônus:

- Gerencie qualquer combinação dos seguintes flags: '`-0.`' e a largura mínima de campo em todas as conversões.
- Gerencie todos os seguintes flags: '`# +`' (Sim, um deles é um espaço)

Se você planeja completar a parte bônus, considere a implementação de seus recursos adicionais desde o início. Isso ajudará você a evitar as armadilhas de uma abordagem ingênua.

A parte bônus só será avaliada se a parte obrigatória estiver PERFEITA. Para ser considerada perfeita, a parte obrigatória deve estar totalmente implementada e funcionar corretamente sem nenhum erro. Se você não passou em TODOS os requisitos obrigatórios, sua parte bônus não será avaliada.

Capítulo VI

Submissão e Avaliação por Pares

Submeta sua tarefa em seu repositório `Git` como de costume. Apenas o trabalho dentro do seu repositório será avaliado durante a defesa. Não hesite em verificar duas vezes os nomes de seus arquivos para garantir que estejam corretos.

Assim que você completar este projeto, você terá permissão para adicionar sua `ft_printf()` à sua `libft`, permitindo seu uso em seus projetos em `C`.

Durante a avaliação, uma breve **modificação do projeto** pode ser ocasionalmente solicitada. Isso pode envolver uma pequena mudança de comportamento, algumas linhas de código para escrever ou reescrever, ou um recurso fácil de adicionar.

Embora esta etapa possa **não ser aplicável a todos os projetos**, você deve estar preparado para ela se for mencionada nas diretrizes de avaliação.

Esta etapa visa verificar sua compreensão real de uma parte específica do projeto. A modificação pode ser realizada em qualquer ambiente de desenvolvimento que você escolher (por exemplo, sua configuração usual), e deve ser viável em poucos minutos — a menos que um prazo específico seja definido como parte da avaliação.
Você pode, por exemplo, ser solicitado a fazer uma pequena atualização em uma função ou script, modificar uma exibição ou ajustar uma estrutura de dados para armazenar novas informações, etc.

Os detalhes (escopo, alvo, etc.) serão especificados nas **diretrizes de avaliação** e podem variar de uma avaliação para outra para o mesmo projeto.

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