

Turbo 4.2

Miniprojeto 01: match / nmatch

Sumário: Segundo mini-projeto do Turbo 4.2.

Conteúdo

I Preâmbulo

II Instruções

III match

IV nmatch

2

3

5

6

Capítulo I

Preâmbulo

Está querendo ficar acordado.... É só seguir as nossas dicas cafeína-free e você vai aprender a ficar acordado no trabalho!

Arrume suas coisas.

Estudos mostram que fazer uma caminhada de 20 minutos pode aumentar seus níveis de energia e diminuir a fadiga.

Envolva seus ouvidos.

Dê um descanso aos seus olhos.

Se alongue.

Abasteça-se com lanches saudáveis.

Quando nada funcionar, use água fria.

Capítulo II

Instruções

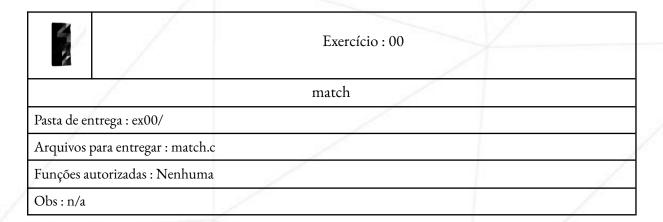
- Somente esta página servirá de referência, não confie nos boatos.
- Atenção! Este documento pode ser alterado até uma hora antes do envio
- Certifique-se de ter as permissões apropriadas em seus arquivos e diretórios.
- Você deve seguir o procedimento de entrega para todos os seus exercícios.
- Os seus exercícios serão corrigidos por seus colegas de curso.
- Além dos seus colegas, haverá a correção de um programa chamado Moulinette.
- A Moulinette é muito rigorosa na sua avaliação. Ela é completamente automatizada. É impossível discutir sua nota com ela. Tenha um rigor exemplar para evitar surpresas.
- A Moulinette não tem a mente muito aberta. Ela não tenta entender o código que não respeita a Norma. A Moulinette utiliza o programa norminette para verificar a norma dos seus arquivos. Então é uma tolice entregar um código que não passa pela norminette.
- A utilização de uma função proibida é um caso de fraude. Qualquer fraude é punida com nota -42.
- Se ft_putchar() for uma função autorizada, compilaremos seu código com nosso ft_putchar.c.
- Você só deve entregar uma função main() se nós pedirmos um programa.
- Esses exercícios são cuidadosamente organizados por ordem de dificuldade do mais fácil ao mais difícil. Não levaremos em consideração um exercício mais difícil concluído com sucesso se um mais fácil não for perfeitamente funcional.
- A Moulinette compila com as sinalizações -Wall -Wextra -Werror, e utiliza gcc.
- Se o seu programa não compila, você terá 0.
- Exercícios em shell devem ser executados com /bin/sh.
- Você não deve deixar em sua pasta nenhum outro arquivo além daqueles

explicitamente especificados pelos enunciados dos exercícios.

- Você tem alguma dúvida? Pergunte ao seu colega da direita. Ou tente também perguntar ao seu colega da esquerda.
- Seu manual de referência se chama Google / man / Internet /
- Examine os exemplos cuidadosamente. Eles podem muito bem pedir detalhes que não são explicitamente mencionados no assunto...
- Por Odin, por Thor! Use seu cérebro!!!

Capítulo III

match

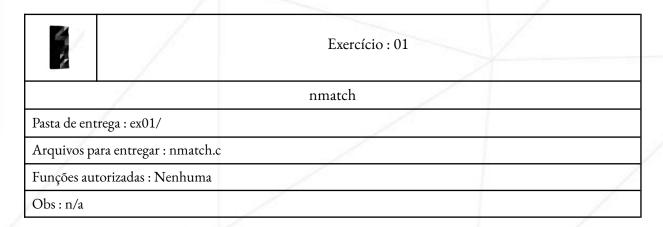


- O propósito dessa função é descobrir se duas strings dão *match* (combinam)
- É considerado que s1 e s2 deem *match* (combinem) quando s1 e s2 forem idênticas
- Se \$2 contém um asterisco ('*'), podemos substituir esse asterisco por qualquer string de caracteres (até vazio) para fazer \$1 e \$2 idênticas.
- s2 pode ter quantos asteriscos você quiser.
- Por exemplo, "main.c" e "*.c" dão *match* (combinam) porque é possível substituir o '*' pela string "main" tornando essas duas strings idênticas.
- É assim que a função deveria ser prototipada:

```
int match(char *s1, char *s2);
```

• A função deve retornar 1 se s1 e s2 derem *match* (combinarem) ou 0 caso contrário

Capítulo IV nmatch



- O objetivo dessa função é contar quantas vezes duas strings dão *match* (combinam).
- Quando nós temos dois ou mais asteriscos, múltiplas combinações de string são possíveis.
- nmatch calcula a quantidade total de combinações
- Seguem alguns exemplos:
 - \circ "abcbd" & "*b*" dão $\it match$ duas vezes: ("a", "cbd") e ("abc", "d")
 - "abc" & "a**" dão match 3 vezes: (nada, "bc"), ("b", "c") and ("bc", nada)
- É assim que a função deve ser prototipada:

```
int nmatch(char *s1, char *s2);
```

• nmatch retorna o número de combinações que dão *match* (combinam)