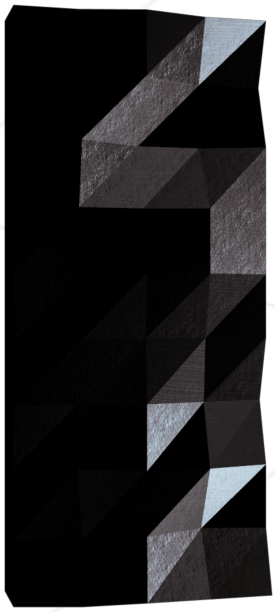

Piscina C

C 02

Sumário: Este documento é o tema do módulo C 02 da Piscina C da 42.

Conteúdo

Capítulo I

Instruções

- Somente esta página servirá de referência, não confie nos boatos.
- Releia bem o tema antes de entregar seus exercícios. A qualquer momento o tema pode mudar.
- Atenção aos direitos de seus arquivos e suas pastas.
- Você deve seguir procedimento de entrega para todos os seus exercícios.
- Os seus exercícios serão corrigidos por seus colegas de piscina.
- 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`.
- Os exercícios estão rigorosamente ordenados do mais simples ao mais complexo. Em nenhum caso daremos atenção, nem levaremos em conta um exercício complexo se outro mais simples não tiver sido perfeitamente realizado.
- A utilização de uma função proibida é um caso de fraude. Qualquer fraude é punida com nota de `-42`.
- Você não deve entregar uma função main() se nós não pedirmos um programa.
- A Moulinette compila com as sinalizações -Wall -Wextra -Werror, e utiliza `gcc`.
- Se o seu programa não compila, você terá `0`.

- 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 vizinho da direita. Ou tente também perguntar ao seu vizinho da esquerda.
- Seu manual de referência se chama `Google / man / Internet / ...`.
- Considere discutir no fórum Piscina do seu Intra, assim como no slack da sua Piscina!
- Leia atentamente os exemplos. Eles podem muito bem pedir coisas que não estão especificadas no tema...
- Reflita. Por favor, por Odin! Por tudo que é mais sagrado.

`Hoje, a Norminette deve ser lançada com a sinalização -R CheckForbiddenSourceHeader. A Moulinette também a utilizará.`

Capítulo II

Preâmbulo

Veja a seguir uma discussão extraída da série `Silicon Valley`:

```
- Por que não usar Vim em vez de Emacs? (RISOS)
- Eu uso Vim em vez de Emac.
- Deus, nos ajude! Está bem. Quer saber? Eu acho que isso
  não vai dar certo. Desculpa. Sabe, como vamos trazer
  filhos ao mundo com eles sabendo disso? Não seria justo
  com eles, você não acha?
- Filhos? Nós ainda nem transamos.
- E adivinha só, agora nunca vamos transar. Porque eu não vou
  ficar com alguém que usa espaços em vez de tabs.
- Richard! (APERTA A BARRA DE ESPAÇO VÁRIAS VEZES)
- Nossa. Tá bom. Adeus.
- Um tab economiza oito espaços! - (BATE A PORTA) - (ESTRONDO)

. . .

(RICHARD GEME DE DOR)

- Meu Deus! Richard, o que aconteceu?
- Eu tentei descer a escada oito degraus de uma vez só. Mas estou bem.
- A gente se vê, Richard.
- Só queria provar o meu argumento.
```

Felizmente, você não precisa utilizar emacs e sua barra de espaço para completar os exercícios.

Capítulo III

Exercício 00 : ft_strcpy

	Exercício : 00
ft_strcpy	
Pasta de entrega : $ex00/$	
Arquivos para entregar : `ft_strcpy.c`	
Funções autorizadas : `Nenhuma`	

- Reproduzir de forma idêntica o funcionamento da função `strcpy` (man strcpy).
- Ela deverá ser prototipada da seguinte maneira:

```
char        *ft_strcpy(char *dest, char *src);
```

Capítulo IV

Exercício 01 : ft_strncpy

	Exercício : 01
ft_strncpy	
Pasta de entrega : $ex01/$	
Arquivos para entregar : `ft_strncpy.c`	
Funções autorizadas : `Nenhuma`	

- Reproduzir de forma idêntica o funcionamento da função `strncpy` (man strncpy).
- Ela deverá ser prototipada da seguinte maneira:

```
char        *ft_strncpy(char *dest, char *src, unsigned int n);
```

Capítulo V

Exercício 02 : ft_str_is_alpha

	Exercício : 02
ft_str_is_alpha	
Pasta de entrega : $ex02/$	
Arquivos para entregar : `ft_str_is_alpha.c`	
Funções autorizadas : `Nenhuma`	

- Escreva uma função que retorne 1 se a string passada como parâmetro só contiver caracteres alfabéticos e retorne 0 se a função contiver outros tipos de caracteres.
- Ela deverá ser prototipada da seguinte maneira:

```
int        ft_str_is_alpha(char *str);
```

- Ela deverá retornar 1 se `str` for uma string vazia.

Capítulo VI

Exercício 03 : ft_str_is_numeric

	Exercício : 03
ft_str_is_numeric	
Pasta de entrega : $ex03/$	
Arquivos para entregar : `ft_str_is_numeric.c`	
Funções autorizadas : `Nenhuma`	

- Escreva uma função que retorne 1 se a string passada como parâmetro só contiver números e retorne 0 se a função contiver outros tipos de caracteres.
- Ela deverá ser prototipada da seguinte maneira:

```
int        ft_str_is_numeric(char *str);
```

- Ela deverá retornar 1 se `str` for uma string vazia.

Capítulo VII

Exercício 04 : ft_str_is_lowercase

	Exercício : 04
ft_str_is_lowercase	
Pasta de entrega : $ex04/$	
Arquivos para entregar : `ft_str_is_lowercase.c`	
Funções autorizadas : `Nenhuma`	

- Escreva uma função que retorne 1 se a string passada como parâmetro só contiver caracteres alfabéticos minúsculos e retorne 0 se a função contiver outros tipos de caracteres.

- Ela deverá ser prototipada da seguinte maneira:

```
int        ft_str_is_lowercase(char *str);
```

- Ela deverá retornar 1 se `str` for uma string vazia.

Capítulo VIII

Exercício 05 : ft_str_is_uppercase

	Exercício : 05
ft_str_is_uppercase	
Pasta de entrega : $ex05/$	
Arquivos para entregar : `ft_str_is_uppercase.c`	
Funções autorizadas : `Nenhuma`	

- Escreva uma função que retorne 1 se a string passada como parâmetro só contiver caracteres alfabéticos maiúsculos e retorne 0 se a função contiver outros tipos de caracteres.

- Ela deverá ser prototipada da seguinte maneira:

```
int        ft_str_is_uppercase(char *str);
```

- Ela deverá retornar 1 se `str` for uma string vazia.

Capítulo IX

Exercício 06 : ft_str_is_printable

	Exercício : 06
ft_str_is_printable	
Pasta de entrega : $ex06/$	
Arquivos para entregar : `ft_str_is_printable.c`	
Funções autorizadas : `Nenhuma`	

- Escreva uma função que retorne 1 se a string passada como parâmetro só contiver caracteres imprimíveis e retorne 0 se a função contiver outros tipos de caracteres.
- Ela deverá ser prototipada da seguinte maneira:

```
int        ft_str_is_printable(char *str);
```

- Ela deverá retornar 1 se `str` for uma string vazia.

Capítulo X

Exercício 07 : ft_strupcase

	Exercício : 07
ft_strupcase	
Pasta de entrega : $ex07/$	
Arquivos para entregar : `ft_strupcase.c`	
Funções autorizadas : `Nenhuma`	

- Escreva uma função que deixe todas as letras em maiúsculo.
- Ela deverá ser prototipada da seguinte maneira:

```
char        *ft_strupcase(char *str);
```

- Ela deverá retornar `str`.

Capítulo XI

Exercício 08 : ft_strlowcase

	Exercício : 08
ft_strlowcase	
Pasta de entrega : $ex08/$	
Arquivos para entregar : `ft_strlowcase.c`	
Funções autorizadas : `Nenhuma`	

- Escreva uma função que deixe todas as letras em minúsculo.
- Ela deverá ser prototipada da seguinte maneira:

```
char	*ft_strlowcase(char *str);
```

- Ela deverá retornar `str`.

Capítulo XII

Exercício 09 : ft_strcapitalize

	Exercício : 09
ft_strcapitalize	
Pasta de entrega : $ex09/$	
Arquivos para entregar : `ft_strcapitalize.c`	
Funções autorizadas : `Nenhuma`	

- Escreva uma função que deixe a primeira letra de cada palavra em maiúsculo e o resto da palavra em minúsculo.
- Uma palavra é uma sequência de caracteres alfanuméricos.
- Ela deverá ser prototipada da seguinte maneira:

```
char        *ft_strcapitalize(char *str);
```

- Ela deverá retornar `str`.
- Por exemplo:

```
oi, tudo bem? 42palavras quarenta-e-duas; cinquenta+e+um
```

- Deve resultar:

```
Oi, Tudo Bem? 42palavras Quarenta-E-Duas; Cinquenta+E+Um
```

Capítulo XIII

Exercício 10 : ft_strlcpy

	Exercício : 10
ft_strlcpy	
Pasta de entrega : $ex10/$	
Arquivos para entregar : `ft_strlcpy.c`	
Funções autorizadas : `Nenhuma`	

- Reproduzir de forma idêntica o funcionamento da função `strlcpy` (man strlcpy).
- Ela deverá ser prototipada da seguinte maneira:

```
unsigned int ft_strlcpy(char *dest, char *src, unsigned int size);
```

Para testar a strlcpy original, você deve compilar o exercício com a flag -lbsd, ex: gcc -Wall -Werror -Wextra -lbsd test.c

Capítulo XIV

Exercício 11 : ft_putstr_non_printable

	Exercício : 11
ft_putstr_with_non_printable	
Pasta de entrega : $ex11/$	
Arquivos para entregar : `ft_putstr_non_printable.c`	
Funções autorizadas : `write`	

- Escreva uma função que mostre uma string de caracteres na tela. Se essa string contiver caracteres não imprimíveis, eles devem ser mostrados na forma hexadecimal (em minúsculo) precedidos por um "backslash".
- Por exemplo, com este parâmetro:

```
Oi\nvoce esta bem?
```

- A função deverá mostrar:

```
Oi\0avoce esta bem?
```

- Ela deverá ser prototipada da seguinte maneira:

```
void        ft_putstr_non_printable(char *str);
```

Capítulo XV

Exercício 12 : ft_print_memory

	Exercício : 12
ft_print_memory	
Pasta de entrega : $ex12/$	
Arquivos para entregar : `ft_print_memory.c`	
Funções autorizadas : `write`	

- Escreva uma função que mostre uma zona de memória na tela.
- A exibição da zona de memória deve estar dividida em três colunas separadas por um espaço:
  - O endereço em hexadecimal do primeiro caractere da linha seguido por um ':'.
  - O conteúdo em hexadecimal com um espaço nos dois caracteres e deve ser completado com espaços se necessário (veja o exemplo abaixo).
  - O conteúdo em caracteres imprimíveis.
- Se um caractere for não imprimível, deve ser substituído por um ponto.
- Cada linha deve administrar dezesseis caracteres.
- Se `size` for igual a 0, nada será mostrado.

- Exemplo:

```
$> ./ft_print_memory
00000010a161f40: 426f 6e6a 6f75 7220 6c65 7320 616d 696e Bonjour les amin
00000010a161f50: 6368 6573 090a 0963 2020 6573 7420 666f ches...c est fo
00000010a161f60: 7509 746f 7574 0963 6520 7175 206f 6e20 u.tout.ce qu on
00000010a161f70: 7065 7574 2066 6169 7265 2061 7665 6309 peut faire avec.
00000010a161f80: 0a09 7072 696e 745f 6d65 6d6f 7279 0a0a ..print_memory..
00000010a161f90: 0a09 6c6f 6c2e 6c6f 6c0a 2000      ..lol.lol. .
$> ./ft_print_memory | cat -te
000000107ff9f40: 426f 6e6a 6f75 7220 6c65 7320 616d 696e Bonjour les amin$
000000107ff9f50: 6368 6573 090a 0963 2020 6573 7420 666f ches...c est fo$
000000107ff9f60: 7509 746f 7574 0963 6520 7175 206f 6e20 u.tout.ce qu on $
000000107ff9f70: 7065 7574 2066 6169 7265 2061 7665 6309 peut faire avec.$
000000107ff9f80: 0a09 7072 696e 745f 6d65 6d6f 7279 0a0a ..print_memory..$
000000107ff9f90: 0a09 6c6f 6c2e 6c6f 6c0a 2000      ..lol.lol. .$
$>
```

- Ela deverá ser prototipada da seguinte maneira:

```
void        *ft_print_memory(void *addr, unsigned int size);
```

- Ela deverá retornar `addr`.