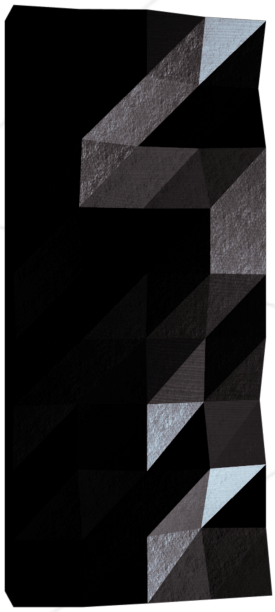

Piscina C

C 02

Sumário: ESTE documento é o enunciado do módulo C 02 da Piscina C da 42.

Versão: 5.4

Conteúdo

Capítulo I

Instruções

- Somente este documento servirá de referência; não confie nos boatos.
- Releia bem o enunciado antes de entregar os seus exercícios. A qualquer momento pode haver alterações.
- Tenha atenção aos direitos dos seus ficheiros e pastas.
- Deverá seguir o procedimento de entrega para todos os exercícios.
- Os seus exercícios serão corrigidos pelos seus colegas de piscine.
- Além dos seus colegas, a Moulinette também corrigirá os seus exercícios.
- A Moulinette é extremamente rígida na sua avaliação. É completamente automatizada, e é impossível discutir a sua nota com ela. Portanto, seja rigoroso!
- A Moulinette não tem uma mente muito aberta: não tenta entender código que não respeita a Norma. A Moulinette utiliza o programa `norminette` para verificar a norma dos ficheiros. Seria uma tontice entregar código que não passa pela `norminette`...
- Os exercícios são ordenados precisamente do mais simples ao mais complexo. Em caso algum consideraremos um exercício mais complexo se outro mais simples não tiver sido perfeitamente realizado.
- A utilização de qualquer função proibida é um caso de fraude. Qualquer fraude é punida com nota de `-42`.
- Deve entregar uma função main() se for pedido um programa.
- A Moulinette compila com as textitflags -Wall -Wextra -Werror, e utiliza `cc`.
- Se o seu programa não compila, terá `0`.
- 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...
- Não deve deixar no repositório de entrega nenhum outro ficheiro além daqueles explicitamente especificados pelo enunciado dos exercícios.
- Leia atentamente os exemplos: podem demonstrar coisas que não estão especificadas no enunciado...

`A Norminette deve ser lançada com a` *`flag`* `-R CheckForbiddenSourceHeader. A Moulinette também a utilizará.`

Capítulo II

Preâmbulo

Here is a discuss extract from the `Silicon Valley` serie:

```
- I mean, why not just use Vim over Emacs? (CHUCKLES)
- I do use Vim over Emac.
- Oh, God, help us! Okay, uh you know what? I just don't think this is
  going to work. I'm so sorry. Uh, I mean like, what, we're going to bring
  kids into this world with that over their heads? That's not really fair
  to them, don't you think?
- Kids? We haven't even slept together.
- And guess what, it's never going to happen now, because there is no
  way I'm going to be with someone who uses spaces over tabs.
- Richard! (PRESS SPACE BAR MANY TIMES)
- Wow. Okay. Goodbye.
- One tab saves you eight spaces! - (DOOR SLAMS) - (BANGING)

. . .

(RICHARD MOANS)

- Oh, my God! Richard, what happened?
- I just tried to go down the stairs eight steps at a time. I'm okay, though.
- See you around, Richard.
- Just making a point.
```

Hopefully, you are not forced to use emacs and your space bar to complete the following exercices.

Nivel Minimo

O limiar de validação para este projeto é de 50%.

Cabe a ti determinar qual exercício te permite atingir esse limiar e se desejas completar mais exercícios.

Capítulo III

Exercício 00 : ft_strcpy

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

- Reproduzir de forma idêntica o funcionamento da função `strcpy` (man strcpy).
- 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 : $ex01/$	
Ficheiros para entregar : `ft_strncpy.c`	
Funções autorizadas : `Nenhuma`	

- Reproduzir de forma idêntica o funcionamento da função `strncpy` (man strncpy).
- 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 : $ex02/$	
Ficheiros 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.
- Deverá ser prototipada da seguinte maneira:

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

- Dverá 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 : $ex03/$	
Ficheiros 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.
- Deverá ser prototipada da seguinte maneira:

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

- 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 : $ex04/$	
Ficheiros 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.

- Deverá ser prototipada da seguinte maneira:

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

- 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 : $ex05/$	
Ficheiros 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.

- Deverá ser prototipada da seguinte maneira:

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

- 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 : $ex06/$	
Ficheiros 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.
- Deverá ser prototipada da seguinte maneira:

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

- Deverá retornar 1 se `str` for uma string vazia.

Capítulo X

Exercício 07 : ft_strupcase

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

- Escreva uma função que coloque todas as letras em maiúsculo.
- Deverá ser prototipada da seguinte maneira:

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

- Deverá retornar `str`.

Capítulo XI

Exercício 08 : ft_strlowcase

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

- Escreva uma função que coloque todas as letras em minúsculo.
- Deverá ser prototipada da seguinte maneira:

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

- Deverá retornar `str`.

Capítulo XII

Exercício 09 : ft_strcapitalize

	Exercício : 09
ft_strcapitalize	
Pasta de entrega : $ex09/$	
Ficheiros 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.
- Deverá ser prototipada da seguinte maneira:

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

- Deverá retornar `str`.
- Por exemplo:

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

- Deve resultar em:

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

Capítulo XIII

Exercício 10 : ft_strlcpy

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

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

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

Capítulo XIV

Exercício 11 : ft_putstr_non_printable

	Exercício : 11
ft_putstr_with_non_printable	
Pasta de entrega : $ex11/$	
Ficheiros 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:

```
Ola\nesta bem?
```

- A função deverá mostrar:

```
Ola\0aesta bem?
```

- 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 : $ex12/$	
Ficheiros 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 primeira 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 tratar de dezesseis caracteres.
- Se `size` for igual a 0, nada será mostrado.

- Exemplo:

```
$> ./ft_print_memory
000000010a161f40: 426f 6e6a 6f75 7220 6c65 7320 616d 696e Bonjour les amin
000000010a161f50: 6368 6573 090a 0963 0720 6573 7420 666f ches...c. est fo
000000010a161f60: 7509 746f 7574 0963 6520 7175 206f 6e20 u.tout.ce qu on
000000010a161f70: 7065 7574 2066 6169 7265 2061 7665 6309 peut faire avec.
000000010a161f80: 0a09 7072 696e 745f 6d65 6d6f 7279 0a0a ..print_memory..
000000010a161f90: 0a09 6c6f 6c2e 6c6f 6c0a 2000           ..lol.lol. .
$> ./ft_print_memory | cat -te
0000000107ff9f40: 426f 6e6a 6f75 7220 6c65 7320 616d 696e Bonjour les amin$
0000000107ff9f50: 6368 6573 090a 0963 0720 6573 7420 666f ches...c. est fo$
0000000107ff9f60: 7509 746f 7574 0963 6520 7175 206f 6e20 u.tout.ce qu on $
0000000107ff9f70: 7065 7574 2066 6169 7265 2061 7665 6309 peut faire avec.$
0000000107ff9f80: 0a09 7072 696e 745f 6d65 6d6f 7279 0a0a ..print_memory..$
0000000107ff9f90: 0a09 6c6f 6c2e 6c6f 6c0a 2000           ..lol.lol. .$
$>
```

- Deverá ser prototipada da seguinte maneira:

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

- Deverá retornar `addr`.

Capítulo XVI

Submissão e avaliação

Entrega o teu trabalho no teu repositório `Git`, como habitual. Apenas o trabalho dentro do teu repositório será avaliado durante a defesa. Não hesites em confirmar os nomes dos teus ficheiros para ter a certeza que estão corretos.

`Apenas precisas de entregar os ficheiros pedidos no enunciado deste projeto.`