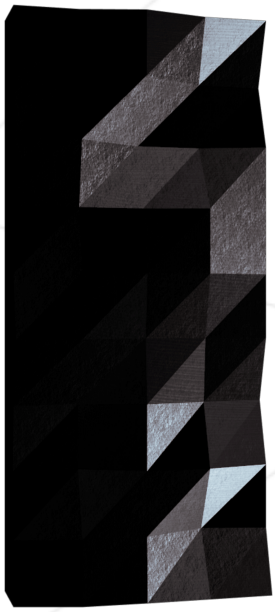

Piscine Reloaded

É bom estar de volta

Summary:
A Piscina foi boa, mas o tempo passou. Esta série de exercícios ajudará você a relembrar dos fundamentos que você aprendeu durante a Piscina. Funções, loops, ponteiros, estruturas, vamos lembrar juntos das bases sintáticas e semânticas do `C`

Version: 1.1

Contents

Chapter I

Preâmbulo

Edward Joseph Snowden (nascido em 21 de junho de 1983) é um profissional da computação americano, ex-funcionário da CIA, e ex-consultor para o Governo dos Estados Unidos que copiou e vazou informações confidenciais da Agência de Segurança Nacional (NSA) em 2013 sem autorização. Seus vazamentos revelaram vários programas de vigilância global, muitos executados pela NSA e pela Five Eyes Intelligence Alliance com a cooperação de empresas de telecomunicações e governos Europeus.

Em 2013, Snowden foi contratado por um empreiteiro da NSA, Booz Allen Hamilton, após ter sido empregado da Dell da CIA. Em 20 de maio de 2013, Snowden voou para Hong Kong depois de deixar seu emprego nas instalações da NSA no Hawaii, e, no início de junho, revelou milhares de documentos confidenciais aos jornalistas Glenn Greenwald, Laura Poitras, e Ewen MacAskill. Snowden chamou a atenção internacional depois que histórias baseadas no material foram publicadas no The Guardian e no The Washington Post. Outras divulgações foram feitas por outras publicações, incluindo Der Spiegel e The New York Times.

Em 21 de junho de 2013, o Departamento de Justiça dos EUA abriu processo contra Snowden de duas acusações de violação da Lei de Espionagem de 1917 e roubo de propriedade do governo. Dois dias depois, ele voou para o aeroporto Sheremetyevo de Moscou, mas as autoridades russas observaram que seu passaporte americano havia sido cancelado e ele ficou restrito ao terminal do aeroporto por mais de um mês. A Rússia finalmente concedeu a ele o direito de asilo por um ano, e repetidas prorrogações permitiram que ele ficasse pelo menos até 2020. Ele supostamente mora em um local não revelado em Moscou e continua buscando asilo em outras partes do mundo.

Um ponto de controvérsia, Snowden foi chamado de herói, denunciante, dissidente, traidor e patriota. Suas revelações alimentaram debates sobre vigilância em have fueled debates over mass surveillance, government secrecy, and the massa, sigilo governamental e o equilíbrio entre segurança nacional e privacidade da informação.

Há um bom documentário na `HBO` aqui.

Chapter II

Introdução

O `Piscine Reloaded` é um compilado dos melhores exercícios que você fez durante a `Piscina C` para relembrar de todos os fundamentos da linguagem de programação `C`. Todos os exercícios devem ser feitos integralmente para desbloquear o próximo projeto.

Se você já fez alguns desses exercícios durante a Piscina C, recomendamos não cair na tentação de recuperar seu código antigo. O aprendizado de programação envolve prática e fazer um código existente não ajuda em nada.

Chapter III

Regras Gerais

- Somente esta página servirá de referência. Não acredite em rumores.
- Assegure-se de ter as permissões apropriadas nos seus arquivos e diretórios.
- Você deve seguir os procedimentos de entrega para cada exercício.
- Seus exercícios serão verificados e avaliados por um programa chamado Moulinette.
- A Moulinette é muito meticulosa em sua avaliação do seu trabalho. Ela é completamente automatizada e não há meios de negociar com ela. Então, se você quiser evitar surpresas ruins, seja o mais possível.
- Exercícios em Shell devem ser executáveis com /bin/sh.
- Você não pode deixar nenhum arquivo adicional no seu diretório além daqueles especificados pelo subject.
- Tem uma dúvida? Pergunte para a pessoa da direita. Ou então tente a pessoa da esquerda.
- Seu guia de referência é chamado `Google / man / a Internet / ...`.
- Verifique o servidor no Discord.
- Examine os exemplos cuidadosamente. Eles poderiam muito bem pedir detalhes que não estão mencionados explicitamente no subject...
- A Moulinette não tem a mente muito aberta. Ela não tentará entender o seu código se você não respeitar a Norma. A Moulinette conta com um programa chamado `norminette` para verificar se seus arquivos respeitam a Norma. TL;DR: seria idiota enviar um trabalho que não passa na verificação da `norminette`.
- Usar funções proibidas é considerado trapaça. Trapaceiros levam `-42`, e esta nota não é negociável.
- Você somente precisará submeter uma função main() se pedirmos por um programa.
- A Moulinette compila com as flags: -Wall -Wextra -Werror, e usa `gcc`.
- Se ft_putchar() é uma função autorizada, nós compilaremos seu código com a nossa `ft_putchar.c`.

- Se seu programa não compilar, você receberá nota `0`.
- Por Odin, por Thor! Use seu cérebro!!!

Chapter IV

Exercício 00 : Ah, sim, maaais...

	Exercise 00
Ah, sim, maaais...	
Turn-in directory : *ex*00/	
Files to turn in : `exo.tar`	
Allowed functions : `None`	

- Crie os seguintes arquivos e diretórios. Faça o que for necessário para que quando você use o comando `ls -l` no seu diretório, o output fique assim:

```
$> ls -l
total XX
drwx--xr-x 2 XX XX  XX Jun 1 20:47 test0
-rwx--xr-- 1 XX XX   4 Jun 1 21:46 test1
dr-x---r-- 2 XX XX  XX Jun 1 22:45 test2
-r-----r-- 2 XX XX   1 Jun 1 23:44 test3
-rw-r----x 1 XX XX   2 Jun 1 23:43 test4
-r-----r-- 2 XX XX   1 Jun 1 23:44 test5
lXXXXXXXXX 1 XX XX   5 Jun 1 22:20 test6 -> test0
$>
```

- Sobre as horas, será aceito se o ano for mostrado no caso da data do exercício (1 Jun) estiver ultrapassada por seis meses ou mais.
- Uma vez feito isso, execute `tar -cf exo.tar *` para criar o arquivo a ser entregue.

Não se preocupe com o que você obtiver ao invés de "XX".

Chapter V

Exercício 01 : Z

	Exercise 01
Só os melhores sabem como mostrar Z	
Turn-in directory : $ex01/$	
Files to turn in : `z`	
Allowed functions : `None`	

- Crie um arquivo chamado `z` que retorne "Z", seguido de uma quebra de linha, sempre que o comando `cat` for usado nele.

```
?>cat z
Z
?>
```

Chapter VI

Exercício 02 : clean

	Exercise 02
Turn-in directory : $ex02/$	
Files to turn in : `clean`	
Allowed functions : `None`	

- Num arquivo chamado `clean` coloque a linha de comando que procurará por todos os arquivos - no diretório atual e em seus subdiretórios - com um nome terminado por ~, ou um nome que comece e termine com `#`
- A linha de comando deverá mostrar e apagar todos os arquivos encontrados.
- Somente um comando é permitido: nada de ';' ou '&&' ou outras gambiarras.

```
man find
```

Chapter VII

Exercício 03 : find_sh

	Exercise 03
find_sh.sh	
Turn-in directory : $ex03/$	
Files to turn in : `find_sh.sh`	
Allowed functions : `None`	

- Escreva uma linha de comando que procure por todos os nomes de arquivos terminados com ".sh" (sem as aspas duplas) no diretório atual e todos os seus subdiretórios. Ela deve mostrar somente os nomes dos arquivos sem o .sh.
- Exemplo de output :

```
$>./find_sh.sh | cat -e
find_sh$
file1$
file2$
file3$
$>
```

Chapter VIII

Exercício 04 : MAC

	Exercise 04
MAC.sh	
Turn-in directory : $ex04/$	
Files to turn in : `MAC.sh`	
Allowed functions : `None`	

- Escreva uma linha de comando que mostre os endereços MAC da sua máquina. Cada endereço deve ser seguido de uma quebra de linha.

```
man ifconfig
```

Chapter IX

Exercício 05 : Você consegue criar?

	Exercise 05
Você consegue criar?	
Turn-in directory : $ex05/$	
Files to turn in : `"\?$*'MaRViN'*$?\"`	
Allowed functions : `None`	

- Crie um arquivo contendo somente "42", e NADA mais.
- Seu nome será:

```
"\?$*'MaRViN'*$?\"
```

- Exemplo:

```
$>ls -lRa *MaRV* | cat -e
-rw---xr-- 1 75355 32015 2 Oct 2 12:21 "\?$*'MaRViN'*$?\"$
$>
```

Chapter X

Exercício 06 : ft_print_alphabet

	Exercise 06
ft_print_alphabet	
Turn-in directory : $ex06/$	
Files to turn in : `ft_print_alphabet.c`	
Allowed functions : `ft_putchar`	

- Escreva uma função que mostre o alfabeto em letras minúsculas, em uma única linha, em ordem ascendente, começando pela letra 'a'.
- Ela deve ser prototipada desta forma:

```
void ft_print_alphabet(void);
```

Não se esqueça de prototipar a função ft_putchar no topo do seu arquivo.

Chapter XI

Exercício 07 : ft_print_numbers

	Exercise 07
ft_print_numbers	
Turn-in directory : $ex07/$	
Files to turn in : `ft_print_numbers.c`	
Allowed functions : `ft_putchar`	

- Escreva uma função que mostre todos os dígitos, em uma única linha, em ordem crescente.
- Ela deve ser prototipada desta forma:

```
void ft_print_numbers(void);
```

Não se esqueça de prototipar a função ft_putchar no topo do seu arquivo.

Chapter XII

Exercício 08 : ft_is_negative

	Exercise 08
ft_is_negative	
Turn-in directory : *ex*08/	
Files to turn in : `ft_is_negative.c`	
Allowed functions : `ft_putchar`	

- Escreva uma função que mostre 'N' ou 'P' dependento do sinal do inteiro colocado como parâmetro. Se `n` for negativo, ,mostre 'N'. Se `n` for positivo ou nulo, mostre 'P'.
- Ela deve ser prototipada desta forma:

```
void ft_is_negative(int n);
```

Não se esqueça de prototipar a função ft_putchar no topo do seu arquivo.

Chapter XIII

Exercício 09 : ft_ft

	Exercise 09
ft_ft	
Turn-in directory : $ex09/$	
Files to turn in : `ft_ft.c`	
Allowed functions : `None`	

- Escreva uma função que tome como parâmetro um ponteiro para int, e atribua o valor "42" a esse int.
- Ela deve ser prototipada desta forma:

```
void ft_ft(int *nbr);
```

Chapter XIV

Exercício 10 : ft_swap

	Exercise 10
ft_swap	
Turn-in directory : $ex10/$	
Files to turn in : `ft_swap.c`	
Allowed functions : `None`	

- Escreva uma função que troque o valor de dois inteiros cujos endereços são passados como parâmetro.
- Ela deve ser prototipada desta forma:

```
void ft_swap(int *a, int *b);
```

Chapter XV

Exercício 11 : ft_div_mod

	Exercise 11
ft_div_mod	
Turn-in directory : $ex11/$	
Files to turn in : `ft_div_mod.c`	
Allowed functions : `None`	

- Escreva uma função `ft_div_mod` prototipada desta forma:

```
void ft_div_mod(int a, int b, int *div, int *mod);
```

- Esta função divide os parâmetros `a` por `b` e armazena o resultado no int apontado por `div`. Ela também armazena o resto da divisão de `a` por `b` no int apontado por `mod`.

Chapter XVI

Exercício 12 : ft_iterative_factorial

	Exercise 12
ft_iterative_factorial	
Turn-in directory : $ex12/$	
Files to turn in : `ft_iterative_factorial.c`	
Allowed functions : `None`	

- Escreva uma função iterativa que retorne um número. Este número é o resultado de uma operação fatorial baseada no número dado como parâmetro.
- Se houver erro, a função deve retornar 0.
- Ela deve ser prototipada desta forma:

```
int ft_iterative_factorial(int nb);
```

- Sua função deve retornar seu resultado em menos de dois segundos.

Chapter XVII

Exercício 13 : ft_recursive_factorial

	Exercise 13
ft_recursive_factorial	
Turn-in directory : $ex13/$	
Files to turn in : `ft_recursive_factorial.c`	
Allowed functions : `None`	

- Crie uma função que retorne o fatorial do número dado como parâmetro.
- Se houver erro, a função deve retornar 0.
- Ela deve ser prototipada desta forma:

```
int ft_recursive_factorial(int nb);
```

Chapter XVIII

Exercício 14 : ft_sqrt

	Exercise 14
ft_sqrt	
Turn-in directory : $ex14/$	
Files to turn in : `ft_sqrt.c`	
Allowed functions : `None`	

- Escreva uma função que retorne a raiz quadrada de um número (se ela existir), ou 0 se a raiz quadrada for um número irracional.
- Ela deve ser prototipada desta forma:

```
int ft_sqrt(int nb);
```

- Sua função deve retornar seu resultado em menos de dois segundos.

Chapter XIX

Exercício 15 : ft_putstr

	Exercise 15
ft_putstr	
Turn-in directory : $ex15/$	
Files to turn in : `ft_putstr.c`	
Allowed functions : `ft_putchar`	

- Escreva uma funç ao que mostre uma string de caracteres no output padrão.
- Ela deve ser prototipada desta forma:

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

Chapter XX

Exercício 16 : ft_strlen

	Exercise 16
ft_strlen	
Turn-in directory : $ex16/$	
Files to turn in : `ft_strlen.c`	
Allowed functions : `None`	

- Reproduce the behavior of the function `strlen` (man strlen).
- Ela deve ser prototipada desta forma:

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

Chapter XXI

Exercício 17 : ft_strcmp

	Exercise 17
ft_strcmp	
Turn-in directory : $ex17/$	
Files to turn in : `ft_strcmp.c`	
Allowed functions : `None`	

- Reproduce the behavior of the function `strcmp` (man strcmp).
- Ela deve ser prototipada desta forma:

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

Chapter XXII

Exercício 18 : ft_print_params

	Exercise 18
	ft_print_params
Turn-in directory : $ex18/$	
Files to turn in : `ft_print_params.c`	
Allowed functions : `ft_putchar`	

- Estamos tratando de um program aqui, você deve então ter uma função `main` em seu arquivo `.c`.
- Escreva um programa que mostre os argumentos passados para ele.
- Example :

```
$>./a.out test1 test2 test3
test1
test2
test3
$>
```

Chapter XXIII

Exercício 19 : ft_sort_params

	Exercise 19
ft_sort_params	
Turn-in directory : $ex19/$	
Files to turn in : `ft_sort_params.c`	
Allowed functions : `ft_putchar`	

- Estamos tratando de um program aqui, você deve ter uma função `main` no seu arquivo `.c`.
- Escreva um programa que mostre os argumentos passados para ele organizados em ordem ascii.
- Ele deve mostrar todos os argumentos, exceto `argv[0]`.
- Todos os argumentos devem ter sua própria linha.

Chapter XXIV

Exercício 20 : ft_strdup

	Exercise 20
ft_strdup	
Turn-in directory : $ex20/$	
Files to turn in : `ft_strdup.c`	
Allowed functions : `malloc`	

- Reproduce the behavior of the function `strdup` (man strdup).
- Ela deve ser prototipada desta forma:

```
char *ft_strdup(char *src);
```

Chapter XXV

Exercício 21 : ft_range

	Exercise 21
ft_range	
Turn-in directory : $ex21/$	
Files to turn in : `ft_range.c`	
Allowed functions : `malloc`	

- Escreva uma função `ft_range` que retorne uma array de `ints`. Esta array de `int` deve conter todos os valores entre `min` e `max`.
- `Min` incluído - `max` excluído.
- Ela deve ser prototipada desta forma:

```
int *ft_range(int min, int max);
```

- Se o valor `min`for maior ou igual ao valor de `max`, deve ser retornado um ponteiro nulo.

Chapter XXVI

Exercício 22 : ft_abs.h

	Exercise 22
ft_abs.h	
Turn-in directory : *ex*22/	
Files to turn in : `ft_abs.h`	
Allowed functions : `None`	

- Crie uma `macro ABS` que substitua seu argumento pelo seu valor absoluto:

```
#define ABS(Value)
```

`Estamos pedindo para que seja feito algo que normalmente é banido pela Norma, esta será a única vez que autorizaremos isso.`

Chapter XXVII

Exercício 23 : ft_point.h

	Exercise 23
ft_point.h	
Turn-in directory : $ex23/$	
Files to turn in : `ft_point.h`	
Allowed functions : `None`	

- Crie um arquivo `ft_point.h` que compilará a seguinte main:

```c
#include "ft_point.h"

void  set_point(t_point *point)
{
  point->x = 42;
  point->y = 21;
}

int   main(void)
{
  t_point   point;

  set_point(&point);
  return (0);
}
```

Chapter XXVIII

Exercício 24 : Makefile

	Exercise 24
Makefile	
Turn-in directory : $ex24/$	
Files to turn in : `Makefile`	
Allowed functions : `None`	

- Escreva um `Makefile` que compilará sua `libft.a`.
- O `Makefile` pegará seus arquivos fonte do diretório "srcs".
- O `Makefile` pegará seu cabeçalho do diretório "includes".
- A lib estará na raiz do exercício.
- O `Makefile` também deverá implementar as seguintes regras: `clean`, `fclean` e `re` assim como `all`.
- `fclean` faz o equivalente a make clean e também apaga o binário criado durante o make. `re` faz o equivalente a make fclean seguido por make.
- Nós somente pegaremos seu Makefile e testaremos com nossos arquivos. Para este exercício, somente essas 5 funções obrigatórias deverão ser consideradas: (`ft_putchar`, `ft_putstr`, `ft_strcmp`, `ft_strlen` e `ft_swap`).

`Cuidado com as wildcards!`

Chapter XXIX

Exercício 25 : ft_foreach

	Exercise 25
ft_foreach	
Turn-in directory : $ex25/$	
Files to turn in : `ft_foreach.c`	
Allowed functions : `None`	

- Escreva uma função `ft_foreach` que, para uma array de ints dada, aplica a função em todos os elementos da array. Esta função será aplicada seguindo a ordem da array.
- Ela deve ser prototipada desta forma:

```
void ft_foreach(int *tab, int length, void (*f)(int));
```

- Por exemplo, a função `ft_foreach` pode ser chamado como a seguir para mostrar todos os ints da array:

```
ft_foreach(tab, 1337, &ft_putnbr);
```

Chapter XXX

Exercício 26 : ft_count_if

	Exercise 26
ft_count_if	
Turn-in directory : $ex26/$	
Files to turn in : `ft_count_if.c`	
Allowed functions : `None`	

- Escreva uma função `ft_count_if` que retornará o número de elementos da array que retornam 1, passados para a função `f`.
- Ela deve ser prototipada desta forma:

```
int ft_count_if(char **tab, int (*f)(char*));
```

- A array será delimitada por 0.

Chapter XXXI

Exercício 27 : display_file

	Exercise 27
display_file	
Turn-in directory : $ex27/$	
Files to turn in : `Makefile, e todos os arquivos necessários para o seu programa`	
Allowed functions : `close, open, read, write`	

- Escreva um program chamado `ft_display_file` que mostre, no output padrão, somente o conteúdo do arquivo dado como argumento.
- O diretório de entrega deve ter um `Makefile` com as seguintes regras: `all`, `clean`, `fclean`. O binário se chamará `ft_display_file`.
- A função `malloc` é proibida. Você somente pode fazer esse exercício declarando uma array de tamanho fixo.
- Todos os arquivos passados como argumento serão válidos.
- Mensagens de erro devem ser mostradas em seu output reservado.

```
$> ./ft_display_file
File name missing.
$> ./ft_display_file Makefile
*content of file Makefile*
$> ./ft_display_file Makefile display_file.c
Too many arguments.
$>
```

Chapter XXXII

Entrega e avaliação entre pares

Entregue seu trabalho em seu repositório `GiT` como de costume. Somente o trabalho presente no seu repositório será avaliado pela `Moulinette`.

Lembramos que, excepcionalmente, esse projeto será avaliado somente pela `Moulinette`. Não haverá avaliação entre pares.

A única nota aceitável para este projeto será 100%. Se a `Moulinette` atribuir a você uma nota inferior, você pode tentar fazer melhor clicando no botão `Retry` da sua página do projeto.

Boa sorte e não se esqueça de usar o cabeçalho da 42 nos seus arquivos!