



Piscina C

C 00

Sommario: QUESTO documento tratta il modulo C 00 della Piscina C @ 42.

Indice

I	Istruzioni	2
II	Preambolo	4
III	Esercizio 00 : ft_putchar	5
IV	Esercizio 01 : ft_print_alphabet	6
V	Esercizio 02 : ft_print_reverse_alphabet	7
VI	Esercizio 03 : ft_print_numbers	8
VII	Esercizio 04 : ft_is_negative	9
VIII	Esercizio 05 : ft_print_comb	10
IX	Esercizio 06 : ft_print_comb2	11
X	Esercizio 07 : ft_putnbr	12
XI	Esercizio 08 : ft_print_combn	13
XII	Consegna e valutazione tra pari	14

Capitolo I

Istruzioni

- Fate riferimento solo a questa pagina: non fidatevi delle dicerie.
- Questo documento può subire variazioni prima della scadenza per la presentazione.
- Controllate i permessi dei vostri file e delle vostre cartelle.
- Dovete seguire le procedure di presentazione per tutti gli esercizi.
- I vostri esercizi saranno controllati e valutati dai vostri compagni di corso.
- Moulinette sarà estremamente meticolosa e severa nel valutare il vostro lavoro. Essendo il suo un processo automatico senza possibilità di ricorso, assicuratevi di essere il più precisi possibile al fine di evitare brutte sorprese.
- I vostri esercizi saranno soggetti, oltre alla valutazione tra pari, al controllo e alla valutazione da parte di un programma chiamato Moulinette.
- Moulinette non ha una mentalità aperta. Non proverà a comprendere il vostro codice se non rispetta la Norma. Moulinette utilizza un programma di nome **norminette** per controllare la validità dei vostri file. TL;DR: sarebbe scocco tentare di consegnare un esercizio che non pass il controllo di **norminette**.
- Gli esercizi sono presentati seguendo un ordine di difficoltà crescente. Ai fini della valutazione **NON** si prendono in considerazione gli esercizi se i precedenti non sono stati completati correttamente
- Usare una funzione non autorizzata viene considerato come barare. Chi bara ottiene un **-42** senza possibilità di ricorso.
- Dovrete consegnare una funzione `main()` solo se l'esercizio richiede un programma.
- Moulinette compila per mezzo di `gcc` utilizzando queste flag: `-Wall -Wextra -Werror`.
- Se il vostro programma non compila, il voto sarà 0.
- NON sarà tollerato ALCUN file aggiuntivo nelle cartelle presentate oltre a quelli specificati in questo documento.

- Dubbi o domande? Chiedi a chi si trova alla tua destra, altrimenti a chi si trova alla tua sinistra
- Your reference guide is called `Google / man / the Internet /`
- Date un'occhiata alla sezione Piscina C del forum dell'Intranet.
- Prestate attenzione agli esempi proposti, in quanto potrebbero mostrare dettagli non esplicitamente presentati nel documento...
- Per Odin, Per Thor ! Usate la testa !!!



Norminette va utilizzata con la flag `-R CheckForbiddenSourceHeader`.
Moulinette farà la stessa cosa.

Capitolo II

Preambolo

Cod liver oil is a nutritional supplement derived from liver of cod fish (Gadidae).

As with most fish oils, it has high levels of the omega-3 fatty acids, eicosapentaenoic acid (EPA) and docosahexaenoic acid (DHA). Cod liver oil also contains vitamin A and vitamin D.


It has historically been taken because of its vitamin A and vitamin D content.

It was once commonly given to children, because vitamin D has been shown to prevent rickets and other symptoms of vitamin D deficiency.

Contrary to Cod liver oil, C is good, eat some!

Capitolo III

Esercizio 00 : ft_putchar

	Esercizio 00
	ft_putchar
	Cartella per la consegna : <i>ex00/</i>
	File da consegnare : ft_putchar.c
	Funzioni permesse : write

- Scrivere una funzione che stampi il carattere passato come parametro.
- Il prototipo è il seguente :


```
void ft_putchar(char c);
```

Per stampare il carattere dovrete utilizzare la funzione **write** in questo modo:

```
write(1, &c, 1);
```

Capitolo IV

Esercizio 01 : ft_print_alphabet


	Esercizio 01
	ft_print_alphabet
	Cartella per la consegna : <i>ex01/</i>
	File da consegnare : ft_print_alphabet.c
	Funzioni permesse : write

- Creare una funzione che stampi l'alfabeto in minuscolo, su una sola riga, in ordine crescente, partendo dalla lettera 'a'.
- Il prototipo è il seguente :

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

Capitolo V

Esercizio 02 : ft_print_reverse_alphabet


	Esercizio 02
	ft_print_reverse_alphabet
	Cartella per la consegna : <i>ex02/</i>
	File da consegnare : ft_print_reverse_alphabet.c
	Funzioni permesse : write

- Creare una funzione che stampi l'alfabeto in minuscolo, su una sola riga, in ordine decrescente, partendo dalla lettera 'z'.
- Il prototipo è il seguente :

```
void ft_print_reverse_alphabet(void);
```


Capitolo VI

Esercizio 03 : ft_print_numbers


	Esercizio 03
ft_print_numbers	
Cartella per la consegna : <i>ex03/</i>	
File da consegnare : ft_print_numbers.c	
Funzioni permesse : write	

- Creare una funzione che stampi tutte le cifre, su una sola riga, in ordine crescente
- Il prototipo è il seguente :

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

Capitolo VII

Esercizio 04 : ft_is_negative


	Esercizio 04
	ft_is_negative
	Cartella per la consegna : <i>ex04/</i>
	File da consegnare : ft_is_negative.c
	Funzioni permesse : write

- Creare una funzione che stampi 'N' o 'P' a seconda del segno dell'intero passato come parametro. Se **n** è negativo, stamperà 'N'. Se **n** è positivo o nullo, stamperà 'P'.
- Il prototipo è il seguente ::

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

Capitolo VIII

Esercizio 05 : ft_print_comb

	Esercizio 05
	ft_print_comb
	Cartella per la consegna : <i>ex05/</i>
	File da consegnare : ft_print_comb.c
	Funzioni permesse : write

- Creare una funzione che stampi tutte le combinazioni possibili utilizzando tre cifre differenti ognuna maggiore della precedente, il tutto deve presentato in ordine crescente.
- L'output dovrà essere :


```
$>./a.out | cat -e
012, 013, 014, 015, 016, 017, 018, 019, 023, ..., 789$>
```

- 987 non è presente perché 789 già lo è.
- 999 non è presente perché la cifra 9 viene ripetuta.
- Il prototipo è il seguente :

```
void ft_print_comb(void);
```

Capitolo IX

Esercizio 06 : ft_print_comb2

	Esercizio 06
	ft_print_comb2
	Cartella per la consegna : <i>ex06/</i>
	File da consegnare : ft_print_comb2.c
	Funzioni permesse : write

- Creare una funzione che stampi tutte le combinazioni possibili utilizzando due numeri compresi tra 00 e 99, il tutto presentato in ordine crescente.

- L'output dovrà essere :


```
$>./a.out | cat -e
00 01, 00 02, 00 03, 00 04, 00 05, ..., 00 99, 01 02, ..., 97 99, 98 99$>
```

- Il prototipo è il seguente :

```
void ft_print_comb2(void);
```

Capitolo X

Esercizio 07 : ft_putnbr

	Esercizio 07
	ft_putnbr
	Cartella per la consegna : <i>ex07/</i>
	File da consegnare : ft_putnbr.c
	Funzioni permesse : write


- Creare una funzione che stampi il numero passato come parametro. La funzione deve essere in grado di stampare tutti i possibili valori di una variabile di tipo **int**.
- Il prototipo è il seguente :

```
void ft_putnbr(int nb);
```

- Ad esempio:
 - `ft_putnbr(42)` displays "42".

Capitolo XI

Esercizio 08 : ft_print_combn

	Esercizio 08
	ft_print_combn
	Cartella per la consegna : <i>ex08/</i>
	File da consegnare : ft_print_combn.c
	Funzioni permesse : write

- Creare una funzione che stampi tutte le possibili combinazioni utilizzando **n** cifre in ordine crescente.
- **n** sarà tale che : $0 < n < 10$.
- Per **n** = 2, l'output sarà :

```
$>./a.out | cat -e
01, 02, 03, ..., 09, 12, ..., 79, 89$>
```

- Il prototipo è il seguente :

```
void ft_print_combn(int n);
```

Capitolo XII

Consegna e valutazione tra pari

Consegna gli esercizi nella tuo repository `Git` come al solito. Durante la difesa verrà considerato unicamente ciò che si trova all'interno della repository. Assicurati di controllare che i nomi dei tuoi file siano corretti.



Devi consegnare solo i file richiesti da questo documento.