


Capitolo VIII

Esercizio 05 : ft_atoi_base


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

- Scrivere una funzione che converta la porzione iniziale della stringa puntata da str a int.
- str sarà un numero in una base specificata nel secondo parametro.
- A parte per la regola della base, la funzione terrà lo stesso comportamento di ft_atoi.
- Se un argomento non è valido, la funzione restituirà 0. Esempi di argomenti non validi :
 - base è vuota o ha dimensione 1;
 - base contiene due caratteri uguali ;
 - base contiene + o - ;
- Il prototipo è il seguente :

```
int      ft_atoi_base(char *str, char *base);
```

Capitolo IV

Esercizio 01 : ft_list_push_front

	Esercizio 01
	ft_list_push_front
	Cartella per la consegna : ex01/
	File da consegnare : ft_list_push_front.c, ft_list.h
	Funzioni permesse : ft_create_elem

- Creare la funzione `ft_list_push_front` che aggiunge un nuovo elemento di tipo `t_list` all'inizio della lista.
- Deve assegnare come valore a `data` quello dell'argomento dato.
- Se ve ne è la necessità deve aggiornare il puntatore all'inizio della lista.
- Il prototipo è il seguente :

```
void      ft_list_push_front(t_list **begin_list, void *data);
```

Capitolo V

Esercizio 02 : ft_list_size



Esercizio 02

ft_list_size

Cartella per la consegna : *ex02/*

File da consegnare : `ft_list_size.c`, `ft_list.h`

Funzioni permesse : Nessuna

- Creare la funzione `ft_list_size` che restituisce il numero di elementi presenti nella lista.
- Il prototipo è il seguente :

```
int ft_list_size(t_list *begin_list);
```

Capitolo XVII

Esercizio 14 : ft_list_sort

	Esercizio 14
	ft_list_sort
	Cartella per la consegna : ex14/
	File da consegnare : ft_list_sort.c, ft_list.h
	Funzioni permesse : Nessuna

- Creare la funzione `ft_list_sort` che ordina gli elementi di una lista in ordine crescente utilizzando una funzione per comparare i dati di due elementi per volta.
- Il prototipo è il seguente :

```
void ft_list_sort(t_list **begin_list, int (*cmp)());
```

- La funzione puntata da `cmp` sarà utilizzata come segue:


```
(*cmp)(list_ptr->data, list_other_ptr->data);
```



`cmp` potrebbe essere `ft_strcmp`.

Capitolo XV

Esercizio 12 : ft_list_remove_if

	Esercizio 12
	ft_list_remove_if
	Cartella per la consegna : <i>ex12/</i>
	File da consegnare : <i>ft_list_remove_if.c</i> , <i>ft_list.h</i>
	Funzioni permesse : <i>free</i>

- Creare la funzione `ft_list_remove_if` che elimina ogni elemento della lista per cui `cmp` restituisce 0.
- Ogni elemento eliminato dalla lista deve essere liberato usando `free_fct`
- Il prototipo è il seguente :

```
void ft_list_remove_if(t_list **begin_list, void *data_ref, int (*cmp)(), void (*free_fct)(void *))
```

- Le funzioni puntate da `cmp` e da `free_fct` saranno utilizzate come segue :

```
(*cmp)(list_ptr->data, data_ref);  
(*free_fct)(list_ptr->data);
```