Da codice etico unisa http://web.unisa.it/uploads/rescue/41/76/codice-etico-e-di-comportamento-unisa.pdf
A**RT. 43 – VIOLAZIONE DEI DOVERI DEL CODICE - STUDENTI**
1. La violazione delle norme del presente Codice da parte degli studenti può dar luogo a sanzioni disciplinari, ai sensi del Regolamento Studenti dell'Ateneo.
2. Quando siano accertate attività tese a modificare indebitamente l'esito delle prove o impedirne una corretta valutazione, il docente o altro preposto al controllo dispone l'annullamento delle prove medesime e la segnalazione al Rettore ai fini dell'attivazione del procedimento disciplinare ai sensi del Regolamento studenti.

Da Regolamento studenti unisa http://web.unisa.it/uploads/rescue/31/19/reg_studenti_2014_web.pdf
ART. 40 – SANZIONI DISCIPLINARI A CARICO DEGLI STUDENTI
1. Le sanzioni che si possono comminare sono le seguenti:
a) ammonizione;
b) interdizione temporanea da uno o più attività formative;
c) esclusione da uno o più esami o altra forma di verifica di profitto per un periodo fino a sei mesi;
d) sospensione temporanea dall'Università con conseguente perdita delle sessioni di esame.
2. La relativa competenza è attribuita al Senato accademico, fatto salvo il diritto dello studente destinatario del provvedimento di essere ascoltato.
3. L'applicazione delle sanzioni disciplinari deve rispondere a criteri di ragionevolezza ed equità, avuto riguardo alla natura della violazione, allo svolgimento dei fatti e alla valutazione degli elementi di prova. Le sanzioni sono comminate in ordine di gradualità secondo la gravità dei fatti.
4. La sanzione è comminata con decreto rettorale.
5. Tutte le sanzioni disciplinari sono registrate nella carriera scolastica dello studente e vengono conseguentemente trascritte nei fogli di congedo.

Istruzioni per lavorare

Individuare il collegamento "traccia" situato sul desktop e copiarne il contenuto nella cartella "Download" (da creare). Qui sarà possibile editare e compilare il codice.

Istruzioni: compilazione

È richiesto scrivere il *makefile* e che il progetto compili con il comando *make*.
Nel caso non si riesca a realizzare un *makefile* corretto è possibile compilare richiamando da linea di comando *compila.bat*. In tal caso ciò **andrà indicato mettendo un commento nel makefile**; si avrà inoltre una penalizzazione sulla valutazione.

Istruzioni: commenti nel codice

È necessario inserire commenti nel codice prodotto. In particolare, è necessario:

- descrivere la funzione realizzata ed il suo funzionamento con un commento inserito immediatamente prima l'intestazione della funzione; è inoltre necessario scrivere quanto eventualmente richiesto nello specifico dalla traccia.
- commentare i punti più importanti del codice con commenti inline.

Esercizio 1. Si implementi, completando il progetto fornito (Esercizio1), una funzione che, sfruttando la **ricorsione**, dato in input una stringa di caratteri, **stampi a video** tale stringa in ordine invertito e **restituisca** il numero di **caratteri alfanumerici** (numeri e lettere minuscole e maiuscole) della stringa (anche il numero di caratteri deve esse contato mediante la **ricorsione**; N.B. la funzione deve restituire il numero di caratteri alfanumerici, non stamparlo direttamente). Se lo si ritiene utile è possibile utilizzare una struttura dati (tra quelle fornite).
Modificare esclusivamente i file main.c e makefile aggiungendo le funzioni necessarie e completando `int main()`.

Si indichi anche, utilizzando i commenti nel codice, la complessità asintotica della procedura implementata, fornendo una breve giustificazione sulla risposta. **ATTENZIONE:** leggere le note sui commenti e la compilazione nella prima pagina della traccia.

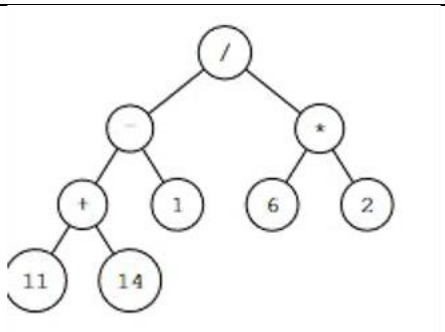
Testare la procedura **leggendo l'input da file input.txt (fornito) dove ogni riga corrisponde ad una stringa da invertire** (NON chiedere input all'utente da tastiera):

- Input: "" (stringa vuota); Output: " 0"
- Input: "b"; Output: "b 1"
- Input: "1"; Output: "1 1"
- Input: "ca"; Output: "ac 1"
- Input: "1a 4n"; Output: "n4 a1 4"
- Input: "Ciao bella gente"; Output: "etneg alleb oaiC 14"
- Input: "Ciao bella gente come va"; Output: "av emoc etneg alleb oaiC 20"
- Input: "Ciao bella gente!!! Come va?"; Output: "?av emoC !!!etneg alleb oaiC 20"
- Input: "Come e' stato bello vederci."; Output: ".icredev olleb otats 'e emoC 22"

Esempio di output: **(NOTA: il vostro programma DEVE riprodurre questo output)**

```
 0
b 1
1 1
ac 2
n4 a1 4
etneg alleb oaiC 14
av emoc etneg alleb oaiC 20
?av emoC !!!etneg alleb oaiC 20
.icredev olleb otats 'e emoC 22
```

Esercizio 2. Si implementi, completando il progetto fornito (Esercizio2), una funzione che, dato un albero binario di input in cui è memorizzata un'espressione aritmetica in forma infissa, restituisca la stringa corrispondente all'espressione originale.
Un albero memorizza la forma infissa di un'espressione nel modo seguente, gli operandi sono inseriti in ordine di apparizione nelle foglie, gli operatori negli altri nodi.

Forma infissa	Forma originale
	((11+14)-1)/(6*2)

Si indichi anche, utilizzando i commenti nel codice, la complessità asintotica della procedura implementata, fornendo una breve giustificazione sulla risposta. **ATTENZIONE:** leggere le note sui commenti e la compilazione nella prima pagina della traccia.

Modificare esclusivamente il file main.c (completando le funzioni già presenti ed eventualmente aggiungendone altre se necessario) **ed il makefile, senza aggiungere o modificare altri file. NON toccare BTree. Soluzioni non realizzate in main.c non verranno prese in considerazione.**

Testare nel main la procedura sui seguenti alberi (NON chiedere input all'utente da tastiera, inserire tutto nel codice C):

- Un albero vuoto;
- Un albero consistente della sola radice contenente il numero 2;
- Un albero instanziato come in figura;
- Un albero casuale (richiamare **newRandomTree(5)**, funzione già presente nel main, per creare l'albero casuale).

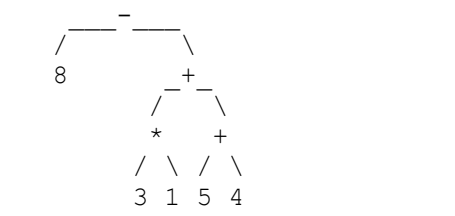
Esempio di output: **(NOTA: il vostro programma DEVE riprodurre questo output, eccetto per l'albero casuale che deve essere differente ad ogni esecuzione del programma)**

```
Albero:
Espressione:

Albero:
2
Espressione: 2

Albero:
      _____/_____
     /           \
    -             *
   / \           / \
  +   1         6   2
 / \
11 14
Espressione: ((11+14)-1)/(6*2)

Albero:
```



Espressione: 8-((3*1)+(5+4))

Esercizio 3. Si implementi, completando il progetto fornito (Esercizio3), una ADT per che modelli il catalogo di una concessionaria di auto usate (dove auto è definita da targa, anno di (re)immatricolazione, modello, anno di produzione), fornendo operatori per:

- inserire una nuova auto nel catalogo;
- rimuovere un'auto dal catalogo;
- ordini il catalogo delle auto per anno di produzione;
- ricerca di un'auto tramite la targa.

Utilizzare la struttura dati più consona per questo tipo di problema.

Si indichi anche, utilizzando i commenti nel codice, la specifica sintattica e semantica degli operatori implementati. **ATTENZIONE:** leggere le note sui commenti e la compilazione nella prima pagina della traccia.

NON modificare i file list.c, list.h, item.h

Completare `main()` in modo da:

1. Aggiungere 4 auto;
2. Rimuove un'auto;
3. Aggiungere un'auto;
4. Ordinare il catalogo;
5. Cercare un'auto presente nel catalogo in base alla targa;
6. Cercare un'auto non presente nel catalogo in base alla targa.

Esempio di output: **(NOTA: il vostro programma DEVE riprodurre questo output)**

```
Aggiungo 4 auto nel catalogo.
Contenuto del catalogo:
targa: JK321UV, immatricolazione: 2015, modello: Audio A4, produzione: 2003
targa: AB123CD, immatricolazione: 2020, modello: FIAT 500, produzione: 2007
targa: XY987ZW, immatricolazione: 2018, modello: BMW M4, produzione: 2014
targa: PQ456LM, immatricolazione: 1999, modello: Opel Corsa, produzione: 1992

-----
Rimozione di: targa: AB123CD, immatricolazione: 2020, modello: FIAT 500,
produzione: 2007
Contenuto del catalogo:
targa: JK321UV, immatricolazione: 2015, modello: Audio A4, produzione: 2003
targa: XY987ZW, immatricolazione: 2018, modello: BMW M4, produzione: 2014
targa: PQ456LM, immatricolazione: 1999, modello: Opel Corsa, produzione: 1992

-----
Aggiunta di: targa: AB123CD, immatricolazione: 2020, modello: FIAT 500,
produzione: 2007
Contenuto del catalogo:
targa: AB123CD, immatricolazione: 2020, modello: FIAT 500, produzione: 2007
targa: JK321UV, immatricolazione: 2015, modello: Audio A4, produzione: 2003
targa: XY987ZW, immatricolazione: 2018, modello: BMW M4, produzione: 2014
targa: PQ456LM, immatricolazione: 1999, modello: Opel Corsa, produzione: 1992

-----
Ordino il catalogo.
Contenuto del catalogo:
```

```
targa: PQ456LM, immatricolazione: 1999, modello: Opel Corsa, produzione: 1992
targa: JK321UV, immatricolazione: 2015, modello: Audio A4, produzione: 2003
targa: AB123CD, immatricolazione: 2020, modello: FIAT 500, produzione: 2007
targa: XY987ZW, immatricolazione: 2018, modello: BMW M4, produzione: 2014

-----
Cerco auto con targa XY987ZW
Trovata:
targa: XY987ZW, immatricolazione: 2018, modello: BMW M4, produzione: 2014

-----
Cerco auto con targa NN000NN
Non trovata.
```