Esame di Programmazione I, 28 febbraio 2014. 2 ore

Esercizio 1 [12 punti] Si scriva una funzione

```
char *replace(const char *where, const char *what) {
```

che restituisce una nuova stringa ottenuta da `where` sostituendo tutti i caratteri `$` con la stringa `what`.

Per esempio, l'esecuzione del seguente `main`:

```
int main(void) {
  const char *s = "ciao $$ amico $";

  char *r = replace(s, "caro");

  printf("%s\n", r);

  free(r);

  return 0;
}
```

dovrà stampare:

```
ciao carocaro amico caro
```

Esercizio 2 [13 punti] Considerando le liste di caratteri come viste a lezione, si scriva una funzione ricorsiva:

```
struct list *replace(struct list *this, struct list *what)
```

che restituisce una lista derivata da `this` sostituendo il carattere `$` con la lista `what`.

Per esempio, il seguente programma:

```
#include <stdlib.h>
#include "list.h"

int main(void) {
  struct list *where = construct_list
    ('d', construct_list('$', construct_list('$', construct_list('E', construct_list('$', NULL)))));
  struct list *what = construct_list('A', construct_list('3', NULL));

  struct list *r = replace(where, what);
  print_list(r);

  return 0;
}
```

dovrà stampare:

```
[d, A, 3, A, 3, E, A, 3]
```

Esercizio 3 [**7 punti**] Si consideri il seguente file di include `cartellone.h` che specifica un cartellone del tipo di quelli delle stazioni ferroviarie, che indica una sequenza di treni specificando il numero e la destinazione di ciascun treno:

```
#ifndef CARTELLONE_H
#define CARTELLONE_H

struct cartellone {
  int numero_treno[8];
  const char *destinazione[8];
  int usati; // il numero di elementi degli array che sono usati
};

struct cartellone *construct_cartellone(void);
void destroy_cartellone(struct cartellone *this);
void print_cartellone(struct cartellone *this);
void aggiungi_treno(struct cartellone *this, int numero_treno, const char *destinazione);
void rimuovi_treno(struct cartellone *this);

#endif
```

Il cartellone nasce vuoto ed è possibile aggiungere treni con la funzione `aggiungi_treno` (specificando il numero del treno e la sua destinazione. Se il cartellone è pieno, questa funzione non fa nulla), stamparlo con la funzione `print_cartellone` e rimuovere il treno che sta in cima al cartellone con la funzione `rimuovi_treno`. Quest'ultima funzione sposta tutti i treni di una posizione verso l'alto.

Se tutto è corretto, l'esecuzione del programma:

```
#include <stdio.h>
#include "cartellone.h"

int main(void) {
  struct cartellone *c = construct_cartellone();

  aggiungi_treno(c, 3408, "Vicenza");
  aggiungi_treno(c, 129, "Milano");
  aggiungi_treno(c, 891, "Roma");

  print_cartellone(c);
  rimuovi_treno(c);  // fa scomparire il treno 3408 per Vicenza
  printf("\n");
  print_cartellone(c);
  destroy_cartellone(c);

  return 0;
}
```

dovrà stampare:

```
3408 - Vicenza
 129 - Milano
 891 - Roma

 129 - Milano
 891 - Roma
```