# Esame di Programmazione I
23 giugno 2014 (tempo disponibile: 2 ore)

## Esercizio 1 (13 punti)

Assumendo che la stringa `s` contenga solo cifre fra 0 e 9, si definisca una funzione

```
char *cifre(const char *s) {
```

che restituisce una nuova stringa contenente la traduzione in italiano delle cifre di `s`.

Se tutto è corretto, l'esecuzione del seguente programma:

```
int main(void) {
  char *temp;
  printf("%s\n", temp = cifre("1234")); free(temp);
  printf("%s\n", temp = cifre("034091")); free(temp);
  printf("%s\n", temp = cifre("")); free(temp);
  printf("%s\n", temp = cifre("9")); free(temp); return 0;
}
```

stamperà:

```
uno due tre quattro
zero tre quattro zero nove uno
                  <- qui c'e' una stringa vuota
nove
```

## Esercizio 2 (8 punti)

Si considerino le liste di `char` come viste a lezione. Si definisca una funzione **ricorsiva**

```
struct list *reflect(struct list *this)
```

che restituisce una lista ottenuta riflettendo `this` rispetto al suo ultimo elemento. La lista `this` non deve essere modificata. Se tutto è corretto, l'esecuzione del programma

```
int main(void) {
  struct list *l = construct_list('a', construct_list('b', construct_list('c', NULL)));
  print_list(reflect(l));
  printf("\n");
  return 0;
}
```

dovrà stampare

```
[a, b, c, b, a]
```

## Esercizio 3 (11 punti)

Si consideri il seguente file `vocabolario.h` che definisce la struttura e le funzioni per un vocabolario italiano/inglese che può contenere fino a 100 voci:

```
#ifndef VOCABOLARIO_H
#define VOCABOLARIO_H
#define SIZE 100

struct voce {     // una singola voce italiano/inglese
  char *italiano;
  char *inglese;
};
```

```
struct vocabolario {  // il vero e proprio vocabolario
  struct voce voci[SIZE];  // le voci contenute nel vocabolario, non tutte usate
  int voci_usate; // quante voci del vocabolario sono gia' state usate
};

struct vocabolario *construct_vocabolario(); // costruisce un vocabolario vuoto
void destroy_vocabolario(struct vocabolario *this); // dealloca un vocabolario
void print_vocabolario(struct vocabolario *this);  // stampa sul video un vocabolario
void add_in_vocabolario(struct vocabolario *this, char *italiano, char *inglese);  // aggiunge una voce al vocabolario
char *cerca_dallitaliano(struct vocabolario *this, char *italiano);  // cerca una traduzione dall'italiano
char *cerca_dallinglese(struct vocabolario *this, char *inglese);  // cerca una traduzione dall'inglese
void sort_italiano(struct vocabolario *this); // ordina il vocabolario rispetto all'italiano
#endif
```

Si completi la seguente implementazione `vocabolario.c`:

```
#include <stdlib.h>
#include <stdio.h>
#include "vocabolario.h"

void print_vocabolario(struct vocabolario *this) {
  int pos;
  printf("%-20s | %-20s\n", "italiano", "inglese");
  printf("%-20s | %-20s\n", "--------", "-------");
  for (pos = 0; pos < this->voci_usate; pos++)
      printf("%-20s | %-20s\n", this->voci[pos].italiano, this->voci[pos].inglese);
}
....
```

Se tutto è corretto, l'esecuzione del seguente programma:

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

int main(void) {
  struct vocabolario *v = construct_vocabolario();
  add_in_vocabolario(v, "cane", "dog"); add_in_vocabolario(v, "topo", "mouse");
  add_in_vocabolario(v, "casa", "house"); add_in_vocabolario(v, "simpatico", "nice");
  add_in_vocabolario(v, "correre", "to run");
  print_vocabolario(v);
  printf("La traduzione in inglese di 'simpatico' e' %s\n", cerca_dallitaliano(v, "simpatico"));
  printf("La traduzione in italiano di 'house' e' %s\n", cerca_dallinglese(v, "house"));
  sort_italiano(v);
  print_vocabolario(v);
  destroy_vocabolario(v); return 0;
}
```

dovrà stampare:

```
italiano             | inglese
--------             | -------
cane                 | dog
topo                 | mouse
casa                 | house
simpatico            | nice
correre              | to run
La traduzione in inglese di 'simpatico' e' nice
La traduzione in italiano di 'house' e' casa
italiano             | inglese
--------             | -------
cane                 | dog
casa                 | house
correre              | to run
simpatico            | nice
topo                 | mouse
```