COGNOME: **NOME:** **MATRICOLA:**

SECONDA PROVA PARZIALE di Programmazione I, 15 Giugno 2015

Esercizio 1 [**11 punti**] Assumendo che la stringa `s` contenga solo cifre da 0 a 9, si definisca una funzione

```
struct lista *cifre(const char *s){ ... }
```

che restituisce una lista di stringhe contenente la traduzione in italiano delle cifre di `s`.
La definizione di `struct lista` é lasciata allo studente.
Inoltre, si definisca un metodo `print` per la stampa della lista.

Se tutto é corretto, l'esecuzione del sequente programma:

```
int main(void){
 char *numeri = "1423";
 struct lista *l = cifre(numeri);
 print(l);
}
```

stamperà

```
uno quattro due tre
```

Infine, si indichi quale particolare tipo di istruzione manca al termine del main dell'esempio di cui sopra.

Esercizio 2 [**11 punti**] Si descriva, con l'ausilio di una figura e di opportuni commenti, lo stato dello stack durante l'esecuzione della seguente funzione ricorsiva, considerando che l'utente inserisca il valore 3:

```
int fattoriale(int n){
  if (n == 0)
   return 1;
  else
   return n * fattoriale(n - 1);
}

int main(void){
  int numero;
  printf("Inserisci un numero non negativo")
  scanf("%d" , &numero);
  printf("il fattoriale di %d e' : %d\n",
   numero,
   fattoriale(numero));
  return 0;
}
```

Esercizio 3 [**11 punti**] Realizzare le seguenti funzioni sulle stringhe:
`str_len()` come la funzione definita nella libreria string, ritorna il numero di caratteri in una stringa.
`str_repeat()` dati una stringa `s` e un intero `n`, crea e ritorna una stringa consistente nella ripetizione di `s` per `n` volte (utilizzare possibilmente la ricorsione).

Se tutto é corretto, l'esecuzione del sequente programma:

```
int main(void){
    printf("%d\n", str_len("ciao"));
    printf("%s\n", str_repeat("ciao ", 3));
}
```

stamperà:

```
4
ciao ciao ciao
```

Soluzione 1

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

struct lista{
 char value;
 struct lista *next;
};

char *translate[] = {"zero", "uno", "due", "tre", "quattro", "cinque", "sei", "sette", "otto", "nove"};

char *to_string(char c){
 return translate[c - '0'];
 /*switch (c){
  case '0':
   return "zero";
  break;
  case '1':
   return "uno";
  break;
  case '2':
   return "due";
  break;
  case '3':
   return "tre";
  break;
  case '4':
   return "quattro";
  break;
  case '5':
   return "cinque";
  break;
  case '6':
   return "sei";
  break;
  case '7':
   return "sette";
  break;
  case '8':
   return "otto";
  break;
  case '9':
   return "nove";
  break;
 }*/
};

//soluzione iterativa
struct lista *cifre(const char *s){
 //if(!s[0]){
 //if(s[0] == '\0'){
 if(!*s){
  return NULL;
 }
 else{
  struct lista *head = (struct lista *)malloc(sizeof(struct lista));
  //head->value = s[0];
```

```
  head->value = *s;
  head->next = NULL;
  struct lista *tail = head;
  //int i = 1;
  //while(s[i] != '\0'){ //while(s[i]){
  // tail->next = (struct lista *)malloc(sizeof(struct lista));
  // tail = tail->next;
  // tail->value = s[i];
  // i++;
  //}
  while(*++s){
   tail->next = (struct lista *)malloc(sizeof(struct lista));
   tail = tail->next;
   tail->value = *s;
  }
  tail->next = NULL;
  return head;
 }
};

//soluzione ricorsiva
struct lista *rec_cifre(const char *s){
 if(!*s){
  return NULL;
 }
 else{
  struct lista *this = (struct lista *)malloc(sizeof(struct lista));
  this->value = *s;
  this->next = rec_cifre(s+1);
  return this;
 }
};

//soluzione ricorsiva 2
struct lista *construct_l(char s, struct lista *next){
 struct lista *this = (struct lista *)malloc(sizeof(struct lista));
 this->value = s;
 this->next = next;
};

struct lista *rec2_cifre(const char *s){
 if(!*s){
  return NULL;
 }
 else{
  return construct_l(*s, rec2_cifre(s+1));
 }
};

void print(struct lista *l){
 //struct lista *c = l;
 //while(c != NULL){
 // printf("%s ", to_string(c->value));
 // c = c->next;
 //}
 //printf("\n");
 while(l != NULL){
```

```
  printf("%s ", to_string(l->value));
  l = l->next;
 }
 printf("\n");
};

void destroy(struct lista *l){
 if(l != NULL){
  struct lista *c = l;
  struct lista *p = NULL;
  while(c != NULL){
   if(p != NULL){
    free(p);
   }
   p = c;
   c = c->next;
  }
  free(p);
 }
};

int main(void){
 char *numeri = "1423";
 struct lista *l = cifre(numeri);
 //struct lista *l = rec_cifre(numeri);
 //struct lista *l = rec2_cifre(numeri);
 print(l);
 destroy(l); //da inserire da parte dello studente
};
```

Soluzione 2 Si veda immagine allegata

Soluzione 3

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>

int str_len_i(const char *s);
char *str_repeat_i(const char *s, int n);
int str_len_r(const char *s);
char *str_repeat_r(const char *s, int n);

// str_len() iterativa
int str_len_i(const char *s) {
    const char *p = s;
    /* Loop over the data in s.  */
    while (*p != '\0') {
        p++;
    }
    return (int)(p - s);
}

// str_len() ricorsiva
int str_len_r(const char *s) {
    if (*s == '\0') { // oppure if (*s == NULL)
        return 0;
    } else {
```

```
        return 1 +  str_len_r(++s);
    }
}

// str_repeat() iterativa
char *str_repeat_i(const char *s, int n) {
    int i;
    char *result = malloc(sizeof(char) * str_len_i(s) * n + sizeof(char));
    result[0] = '\0';

    for(i = 0; i < n; i++) {
        strcat(result, s);
    }

    return result;
}

// str_repeat() ricorsiva
char *str_repeat_r(const char *s, int n) {
    if(n == 0) {
        char *result = malloc(1);
        result[0] = '\0';
        return result;
    } else {
        char *result = malloc(sizeof(char) * str_len_r(s) * n + sizeof(char));
        strcpy(result, s);
        return strcat(result, str_repeat_r(s, n - 1));
    }
}
```