Seconda prova parziale di Programmazione I

15 giugno 2012 (tempo disponibile: 2 ore)

Esercizio 1

(19 punti)

Si scrivano i file `polinomio.h` e `polinomio.c` che definiscono un polinomio a coefficienti interi in una sola variabile, cioè una cosa del tipo $-3x^4+9x^3-11x^2-2$. Si devono definire e implementare le seguenti funzioni:

- `struct polinomio *construct_polinomio(int coefficienti[], int grado)`, che costruisce un nuovo polinomio del grado indicato con i coefficienti indicati. Il grado è l'esponente del monomio più significativo. Si noti che il grado è sempre la lunghezza dell'array dei coefficienti meno 1. I coefficienti iniziano con quello del monomio più significativo. Per esempio, per costruire $-3x^4+9x^3-11x^2-2$ si deve poter scrivere `construct_polinomio(coefficienti, 4)` dove `int coefficienti[] = { -3, 9, -11, 0, -2 };`
- `void destruct_polinomio(struct polinomio *this)`, che dealloca il polinomio indicato;
- `int grado(struct polinomio *this)`, che restituisce il grado del polinomio indicato;
- `struct polinomio *add(struct polinomio *this, struct polinomio *other)`, che restituisce un nuovo polinomio che è la somma dei due polinomi indicati;
- `int evaluate(struct polinomio *this, int x)`, che restituisce il valore del polinomio indicato valutato nel punto indicato;
- `char *toString(struct polinomio *this)`, che restituisce una nuova stringa che descrive il polinomio indicato, una cosa del tipo `- 3x^4 + 9x^3 - 11x^2 - 2x^0` (i monomi di coefficiente 0 non devono essere riportati).

Suggerimento: per realizzare la `toString`, può essere comodo ricordarsi che la funzione della libreria standard `sprintf(buffer, format, valori...)` si comporta come `printf(format, valori...)` ma stampa dentro la stringa `buffer` invece che sul video. Inoltre `sprintf` ritorna un `int` che è il numero di caratteri stampati.

Se tutto è corretto, il seguente programma:

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

int main() {
  int coefficienti1[] = { -3, 4, 0, 0, -2 };
  int coefficienti2[] = { 5, -11, 0, 0 };

  struct polinomio *poly1 = construct_polinomio(coefficienti1, 4);
  struct polinomio *poly2 = construct_polinomio(coefficienti2, 3);
  struct polinomio *somma = add(poly1, poly2);
```

```
  char *s;

  printf("primo polinomio: %s di grado %d\n", s = toString(poly1), grado(poly1));
  free(s);

  printf("secondo polinomio: %s di grado %d\n", s = toString(poly2), grado(poly2));
  free(s);

  printf("somma: %s di grado %d\n", s = toString(somma), grado(somma));
  free(s);

  printf("la somma valutata in x = 7 vale %d\n", evaluate(somma, 7));

  destruct_polinomio(somma);
  destruct_polinomio(poly2);
  destruct_polinomio(poly1);

  return 0;
}
```

stamperà:

```
primo polinomio:  - 3x^4 + 4x^3 - 2x^0 di grado 4
secondo polinomio:  + 5x^3 - 11x^2 di grado 3
somma:  - 3x^4 + 9x^3 - 11x^2 - 2x^0 di grado 4
la somma valutata in x = 7 vale -4657
```

Esercizio 2

(13 punti)

Si scriva una funzione ricorsiva `sup` che riceve come argomento una lista `this` (possibilmente vuota) e restituisce una lista contenente solo gli elementi di `this` che sono strettamente maggiori di quelli che li seguono in `this`. La lista `this` non deve essere modificata. Per esempio, se `this` fosse la lista `[5,13,8,9,8,6,2,3,2]` allora `sup(this)` deve essere la lista `[13,9,8,6,3,2]`. Si scriva quindi un programma `main_list.c` con una funzione `main` che costruisce la lista `[5,13,8,9,8,6,2,3,2]`, la stampa con la funzione `print_list`, chiama la funzione `sup` su tale lista e stampa il risultato di `sup` con `print_list`. La definizione delle liste e le funzioni di costruzione e di stampa delle liste sono state viste a lezione: non riscrivetele nella soluzione.