Algoritmi e Strutture Dati 15/16	*Durata: 3 ore*

Prova Scritta del 17 Febbraio 2016

1. Si vuole progettare una struttura dati per *polinomi*, definiti come una sequenza ordinata di coppie $<\text{esponente}, \text{coefficiente}>$. Ad esempio il polinomio $p(x) = x^2 + 3$ corrisponde alla sequenza $\{<0, 3>, <2, 1>\}$. Completare la specifica di *polinomio*, fornendo la specifica semantica per mezzo di pre e post condizioni (specifica costruttiva o modello astratto), rispetto alla seguente specifica sintattica:

   domini: polinomio, coefficiente, esponente

   operatori:

   (a) crea() → polinomio
   // crea un nuovo polinomio di grado 0, $p(x) = 0$

   (b) settaEsponente(polinomio, esponente, coefficiente) → polinomio
   // imposta il coefficiente (ultimo parametro) di un dato esponente (secondo parametro)

   (c) rimuoviEsponente(polinomio, esponente) → polinomio
   // imposta a zero il coefficiente di un dato esponente (secondo parametro)

   (d) leggiCoefficiente(polinomio, esponente) → coefficiente
   // restituisce il coefficiente di un dato esponente (secondo parametro)

   (e) grado(polinomio) → esponente
   // restituisce l'esponente più grande

   (f) somma(polinomio, polinomio) → polinomio
   // restituisce la somma di due polinomi

   (g) prodotto(polinomio, polinomio) → polinomio
   // restituisce il prodotto di due polinomi [7pt]

   SOLUZIONE

   domini: *polinomio*: famiglia di insiemi di coppie di tipo $< c, e >$ dove c è di tipo coefficiente ed e è di tipo esponente; *coefficiente*: insieme dei valori tipo reale; *esponente*: insieme dei valori di tipo intero.

   operatori:

   (a) crea() → p
   - PRE: -
   - POST: p = $\emptyset$

   (b) settaEsponente(p, e, c) → p'
   - PRE: $\nexists < c', e' > \in p$ t.c. $e = e'$
   - POST: $p' = p \cup < c, e >$

   (c) rimuoviEsponente(p, e) → p'
   - PRE: $\exists < c', e' > \in p$ t.c. $e = e'$
   - POST: $p' = p \setminus < c, e >$

   (d) leggiCoefficiente(p, e) → c
   - PRE: $\exists < c', e' > \in p$ t.c. $e = e'$
   - POST: $c = c'$

   (e) grado(p) → e
   - PRE: -
   - POST: se $p = \emptyset$ allora $e = 0$, altrimenti $e = e'$ t.c. $< e', c' > \in p$ e $\nexists < e'', c'' > \in p$ t.c. $e'' > e'$

(f) somma(p, p') $\rightarrow$ p"

- PRE:
- POST: $p'' = \{< c, e >\}$ t.c. $< c, e >$

(g) prodotto(p, p') $\rightarrow$ p"

- PRE:
- POST:

2. Fornire in C++ una possibile realizzazione e rappresentazione della struttura dati polinomio al punto 1), riportando solo la definizione di classe: variabili di classe e definizione dei metodi. Motivare la scelta di altre strutture dati nel caso se ne faccia uso. [4pt]
3. Fornire la specifica sintattica e semantica degli operatori *costrBinAlbero* e *cancSottoBinAlbero* per la struttura dati *Alberi binari* [3pt]
4. Spiegare la realizzazione di alberi binari mediante *vettore* e *collegata con cursori*, fornendo vantaggi e svantaggi di ognuna [4pt]
5. Fornire in pseudocodice l'algoritmo di *visita in post-ordine* per alberi binari [3pt]
6. Descrivere come un albero binario pu essere utilizzato per rappresentare una coda con priorità, descrivendo inoltre come si effettuano le operazioni di inserimento e cancellazione di elementi [6pt]
7. Spiegare cosa è un *problema di ottimizzazione* e quali tecniche si possono adottare per risolverlo [6pt]