Programmazione Strutturata.
Studio di un caso: Ordinamento con QuickSort

Sommario: 19-... marzo, 2015

- Sviluppo Top Down: Specifica del 19/3
- Sviluppo Top Down: Raffinamento del 21/3
- ...

Specifica (iniziale) del 19/3

- Obiettivo Generale: Programma di Ordinamento
- Obiettivo Specifico: Quicksort per sequenze ordinabili
- Vincoli:
  - Una sequenza **c**
  - di valori ordinabili **t**
  - Quindi, **c** è un valore di tipo **Seq(t)**, sequenze di tipo `t`.
- Soluzione/Algoritmo:
  a. Sia **cc** la sequenza corrente da ordinare
  b. Sia **a** di tipo **t**, un elemento di separazione per **cc**
  c. Dividiamo **cc** in due sottosequenze **cLE** e **cGT** separate da **a**
  d. Ripetere [a]-[d] su ogni sottosequenza non singoletta, ottenuta al passo [c] se presente.
  e. Terminiamo, presentando la sequenza ottenuta.

Specifica (iniziale) del 19/3

Procediamo con `N` raffinamenti successivisi, individuando funzionalità e aggiungendo dettagli

- Hints. Chiarire e precisare dettagli rispondendo a domande, quali:
- Cosa si deve intendere per elemento di separazione?
- Cosa si deve intendere per divisione di sequenza rispetto a elemento di separazione, dato un ordinamento?
- ...

Raffinamento del 21/3

- Obiettivo Generale: Programma di Ordinamento
- Obiettivo Specifico: Quicksort per sequenze ordinabili
- Vincoli:
  - Una sequenza **c** di valori ordinabili **t**
  - Sia **Seq(t)** il tipo sequenze di `t`. Allora, **c**:**Seq(t)**[1]
  - Sia $\mathbf{Ord_t}$, l'ordinamento totale considerato. Per ogni $\mathtt{v1} \neq \mathtt{v2}$ di tipo `t`, $(\mathtt{v1}, \mathtt{v2}) \in \mathbf{Ord_t}$ oppure $(\mathtt{v2}, \mathtt{v1}) \in \mathbf{Ord_t}$.
- Soluzione/Algoritmo: *idem*
- Usiamo:
  - Elemento di Separazione. Sia `u`:**Seq(t)**. Sia `Set(u)` l'insieme di tutti i valori che occorrono in `u`. Il separatore di `u` è un valore $\mathtt{v_u} : \mathtt{t}$ tale che, siano:
    $+\mathtt{LE_{u,v_u}} = \{\mathtt{v} \in \mathtt{Set(u)} \mid \mathtt{v} \equiv \mathtt{v_u} \text{ oppure } (\mathtt{v}, \mathtt{v_u}) \in \mathtt{Ord}\}$,
    $+\mathtt{GT_{u,v_u}} = \{\mathtt{v} \in \mathtt{Set(u)} \mid (\mathtt{v_u}, \mathtt{v}) \in \mathtt{Ord}\}$
    Allora, $\mathtt{LE_{u,v_u}} \neq \{\} \neq \mathtt{GT_{u,v_u}}$.

[1] Usiamo la notazione `x:t`, per indicare che `x` è un termine di tipo `t`. Useremo anche, $\mathtt{t1} \times \ldots \times \mathtt{tn} \rightarrow \mathtt{t}$ per indicare un tipo funzione con `n` parametri di tipo `ti` e immagine di tipo `t`.

Raffinamento del 21/3

- Usiamo (continua):
  - Elemento di Separazione (continua)
    - Proprietà. Ogni sequenza ha un elemento di separazione dato, un ordinamento, eccetto che: (a) sia vuota, (b) contenga un solo elemento (anche ripetuto)
  - Sequenza **Seq(t)**. una struttura con le seguenti operazioni:
    - empty: $() \rightarrow Seq(t)$ - seq vuota
    - add: $Seq(t) \times t \rightarrow Seq(t)$ - aggiunta di elemento in coda
    - append: $Seq(t) \times Seq(t) \rightarrow Seq(t)$ - concatenazione di seq
    - size: $Seq(t) \rightarrow int$ - numero elementi
    - at: $Seq(t) \times int \rightarrow t$ - selezione elemento

Raffinamento del 21/3

- Usiamo (continua/2):
  - Divisione di sequenza. Dato $\mathtt{u}$:**Seq(t)**. Dato separatore $\mathtt{v_u} : \mathtt{t}$ per $\mathtt{u}$, dato $\mathbf{Ord_t}$. Siano $\mathtt{QLE_{u,v_u}}$, $\mathtt{QGT_{u,v_u}}$, tali che:
    $+\forall 1 \leq \mathtt{i} \leq \mathtt{size}(\mathtt{QLE_{u,v_u}}),\ \ \mathtt{at}(\mathtt{QLE_{u,v_u}}, \mathtt{i}) \equiv \mathtt{v_u}$ oppure $(\mathtt{at}(\mathtt{QLE_{u,v_u}}, \mathtt{i}), \mathtt{v_u}) \in \mathtt{Ord_t}$
    $+\forall 1 \leq \mathtt{i} \leq \mathtt{size}(\mathtt{QGT_{u,v_u}}),\ \ (\mathtt{v_u}, \mathtt{at}(\mathtt{QLE_{u,v_u}}, \mathtt{i})) \in \mathtt{Ord_t}$
    $+\exists \mathtt{g} \in \mathcal{P}^{\mathtt{size(u)}}$. Sia[2] $\mathtt{u}' = \mathtt{append}(\mathtt{QLE_{u,v_u}}, \mathtt{QGT_{u,v_u}})$. $\mathtt{at}(\mathtt{u}', \mathtt{g(i)}) \equiv \mathtt{at}(\mathtt{u}, \mathtt{i}), \forall 1 \leq \mathtt{i} \leq \mathtt{size(u)}$
    $+\mathtt{size}(\mathtt{QGT_{u,v_u}}) + \mathtt{size}(\mathtt{QLE_{u,v_u}}) = \mathtt{size(u)}$

    Allora, $\mathtt{QLE_{u,v_u}}, \mathtt{QGT_{u,v_u}}$ sono la divisione di $\mathtt{u}$ in sottosequenze separate da $\mathtt{v_u} : \mathtt{t}$, dato $\mathbf{Ord_t}$.

[2] $\mathcal{P}^{\mathtt{k}}$ sono le funzioni di permutazione, i.e. iniettive e suriettive, su $[1..\mathtt{k}]$

Raffinamento del 21/3

Possiamo ritenerci soddisfatti della Specifica Astratta data?

- Il procedimento di calcolo da seguire è completamente definito?
- Sono richiamate, in esso, trasformazioni di applicazione imprecisa o oscura?
- Sono richiamate, in esso, ~~trasformazioni di applicazione~~ questa astrazione è troppo di troppo imprecisa o oscura?
- Se la risposta è SI, NO, NO. Allora Specifica Astratta (SA) è finita e si passa a:
- La progettazione del codice guidati da SA per definire:
  - La struttura (top-level) del programma
  - Le unità di programmazione
  - Le strutture dati

Progettazione del Codice 25/3

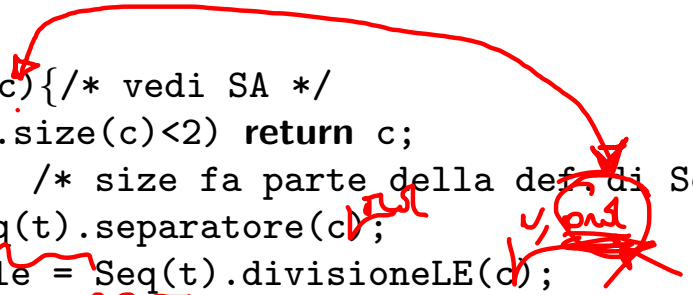
- La struttura (top-level) del programma.
  - Deve definire un ambiente in cui:
    (a) sia definita un'operazione QSort e
    (b) sia possibile applicarla a sequenze ordinabili

```
{....
   type Seq(t) = ....;
   Seq(t) QSort(Seq(t) c) {...};
   ...
/* esempio di uso */
   Seq(int) u = ...;
   u = QSort(Seq(t) u);
}
```

- Completiamo con la notazione usata (non abbiamo ancora scelto un LP)
- Troppo semplice? Raffiniamola completandola a partire dal corso di QSort.

Progettazione del Codice 25/3 - continua/1

```
{....
   type Seq(t) = ....;
   Seq(t) QSort(Seq(t) c){/* vedi SA */
             if (Seq(t).size(c)<2) return c;
                          /* size fa parte della def. di Seq(t) */
             {t v = Seq(t).separatore(c);
              Seq(t) qle = Seq(t).divisioneLE(c);
              Seq(t) qgt = Seq(t).divisioneGT(c);
              return Seq(t).append(QSort(gle),QSort(ggt));
             }
   }
   ...
/* esempio di uso */
   Seq(int) u = ...;
   u = QSort(Seq(t) u);
}
```



- Dobbiamo completare con la definizione di `Seq(t)` trattata come una unità di programmazione per dati.

Progettazione del Codice 25/3 - continua/2

Correzioni nella precedente definizione di QSort:

- L'ordinamento deve essere un parametro di QSort.
- Le definizioni di divisioneLE, divisioneLG, separatore sono locali a QSort e non definite in Seq(t).
- separatore ha anche un p. ordinamento, come divisioneLE e divisioneLG, che hanno inoltre un p. separatore.

```
{....
   type Seq(t) = ....;
   Seq(t) QSort(Seq(t) c, t×t→bool ord){/* vedi SA */
            t separatore(Seq(t) cc, t×t→bool co){...};
            Seq(t) divisioneLE(Seq(t) cc, t cv, t×t→bool co){...};
            Seq(t) divisioneGT(Seq(t) cc, t cv, t×t→bool co){...};
            if (Seq(t).size(c)<2) return c;
                        /* size fa parte della def. di Seq(t) */
            {t a = separatore(c,ord);
             Seq(t) qle = divisioneLE(c,a,ord);
             Seq(t) qgt = divisioneGT(c,a,ord);
             return Seq(t).append(QSort(gle),QSort(ggt));
            }
   }
   ...
/* esempio di uso */
...;
```

Progettazione del Codice 25/3 - continua/3

Cose da fare:

- Completare con la definizione di `Seq(t)` trattata come una unità di programmazione per dati qualunque (definiamolo come modulo `xxx.h` in C, instanziando il parametro di tipo `t` su un tipo a scelta)
- Completare con la definizione di divisioneLE / divisioneGT (pensando all'impiego di un'unica funzione in abbinamento a tipi per valori coppia di sequenze)
- Pensare a un prototipo eseguibile nel linguaggio che conosciamo, ovvero ANSI C.

Prossima discussione Mercoledì 1 aprile ore 14:00-16:00