Esercizi L2 (Algoritmi, Programmi, **LP**) 28/2 - 2/3 2018

Sommario: 28/2-2/3, 2018

1 Quali sono le 4 caratteristiche di un procedimento perchè sia effettivo?

2 Si descriva l'algoritmo del quicksort utilizzato in Esercizio L1.1 e si mostri che esso definisce un procedimento effettivo.

2bis Gli algoritmi forniscono procedimenti effettivi ma non eseguibili per le funzioni calcolabili. Questi procedimenti sono utilizzati, come un canovaccio, per la loro implementazione in programmi che a loro volta, forniscono processi automatici e dunque, eseguibili. Si commentino gli altri 2 principali usi, visti nel corso, di tali procedimenti.

3 La funzione π_1, descritta sotto, è calcolabile.

$$\pi_1(\mathtt{n}) = \begin{cases} 1 & \mathtt{se\ l'espansione\ decimale\ di\ } \pi \mathtt{\ contiene\ n\ 1\ in\ sequenza} \\ 0 & \mathtt{altrimenti} \end{cases}$$

(a) Sapreste fornire una classe di programmi contenente **un** programma per π_1?
(b) Sapreste giustificare l'affermata calcolabilità di π?

4 Si esamini la definizione data alle pagine 509-510 nell'articolo sulla 3-Colorabilità di N. Jonoska et al., citato a lezione e si risponda:
(a) Perchè il procedimento in 4 passi, in pag. 510, definisce un algoritmo?

Sommario: 27/2-2/3, 2018

(b) Sapreste fornire un programma in C che implementi tale algoritmo?
(c) Sapreste fornire un programma in C che introduca una rappresentazione dei flexible tiles ed emuli le operazioni anneal, ligate, cleave, extract?
(d) Discutere la relazione tra la soluzione data in (b) e quella data in (c).

√5 Sia π, la funzione così definita:

$$\pi(\mathtt{n}) = \begin{cases} 1 & \texttt{se l'espansione decimale di } \pi \texttt{ contiene n} \\ 0 & \texttt{altrimenti} \end{cases}$$

(a) Sapreste fornire una classe di programmi nei quali è certamente contenuto anche **un** programma che calcola π?
(b) Sapreste fornire argomenti per affermare, o per negare la calcolabilità di π?

6 SmallC è un sotto-linguaggio di C senza procedure e costrutti correlati. Si scriva in SmallC un programma che calcoli la funzione *ordinamento di sequenza*.

6bis SmallC è un sotto-linguaggio di C che mantiene tutti i costrutti C fatta eccezione per le procedure e costrutti correlati. Si può scrivere in SmallC un programma che implementa l'algoritmo di quicksort? Si giustifichi la risposta adeguatamente.

Esercizi L2 (Principi di Calcolabilità) 28/2 - 2/3 2018 /3

Sommario: 27/2-2/3, 2018

√6ter SmallC è un sotto-linguaggio di C che mantiene tutti i costrutti C fatta eccezione per le procedure e costrutti correlati. Si può dimostrare che SmallC∈**LP**. Si dica cosa implica tale fatto?

7 L'algortitmo di quicksort definito originariamente da Hoare non pone vincoli sulla: a) dimensione finita della sequenza considerata, nè sul b) tipo di valori formanti la sequenza, nè sulla c) relazione di ordinamento applicata. Di modo che lo stesso algoritmo è applicabile ad ogni sequenza di valori rappresentabili nel linguaggio e su cui sia possibile definire una relazione di ordinamento (anche parziale). Quali di questi vincoli sono presenti nella soluzione data all' Esercizio L1.1 di questo corso?

8 (a) Quali tra i seguenti linguaggi sono tra loro collegati: ADA, JAVA, Haskell, SQL, Miranda, Lisp, Smalltalk, ML, C++, Pascal, OCAML?
(b) Per ogni gruppo individuato in (a) si indica natura del collegamento e fonte dell'affermazione.

9 Nella diffusione dei linguaggi, richiamata a lezione, sono considerate 4 tipologie di applicazioni.
(a) Si elenchino.
(b) Per ciascuna tipologia si menzioni almeno un'applicazione ed un link web

Sommario: 27/2-2/3, 2018

10 Si consideri il programma QSort2, allegato nei listings della settimana, Listing1, proposto come "soluzione" dell'esercizio L1.1. Si rifrasino le costruzioni utilizzate in modo da completare il programma QSort3 sotto, equivalente a QSort2.

```
int N = ...; //Parametro di programma (costante, non modificabile)
void Scambia(...){...}
void LeggiSeq(...){...}
void StampaSeq(...){...}
void QuickSort(int Left, int Right, int A[]){...}

int main(void){
    int Seq[N];
    int k;
    LeggiSeq(Seq,&k);//Lettura sequenza da ordinare
    QuickSort(Seq,k);//Ordinamento
    StampaSeq(Seq,k);//Stampa sequenza ordinata
    return(1);
}
```

11 I programmi QSort2 e QSort3 in esercizi L1.1 e L2.10 sono equivalenti e interscambiabili dal punto di vista della funzione calcolata. Ben diverso è invece dal punto di vista dell'espressività. Facendo riferimento a quanto visto a lezione, si dica:
(a) In cosa differiscono QSort2 e QSort3;

Sommario: 27/2-2/3, 2018

(b) Quale dei due è più espressivo;
(c) Quale dei due è più prossimo, e perchè, ad un'implementazione dell'algoritmo di Hoare, richiamato in esercizio L2.7

12 (a) Si discuta una modifica del programma incompleto di esercizio L2.10, sopra, tale che il nuovo programma possa ordinare in accordo ad un ordinamento g su interi, totale ma arbitrario.
(b) Si completi il programma ottenuto in (a) in modo da ottenere un programma QSort4 equivalente a QSort2 e QSort3. Si evidenzi definizione e uso di g.
(c) Si compili QSort4 e si esegua il sorgente con sequenza di imput `3, -7, 5, -3, 7`, opportunamente codificata in accordo alle assunzioni fatte sul formato di I/O del programma
(d) Si modifichi QSort4 in un programma che utilizzi un ordinamento: $g(n, m)$ sse $|n| \geq |m|$. In particolare, si mostri la nuova definizione di g, si compili ed esegua il nuovo programma con la sequenza introdotta in (c).

13 (a) Sapreste scrivere un comando, che chiameremo NOP, in C che aggiunto ad un programma in C non modifica la funzione calcolata da tale programma.
(b) Quale funzione calcola un programma C costituito dal solo comando NOP?
(c) Quanti programmi C calcolano una stessa funzione calcolabile? Si motivi la risposta.

Esercizi L2 - 28/2/2018

Esercizio (1)

Quali sono le 4 caratteristiche che deve avere un procedimento perchè sia effettivo?

Soluzione

Esercizi L2 - 28/2/2018

Esercizio (2)

(a) Si fornisca l'algoritmo del quicksort implementato dal programma dato come soluzione di Esercizio L1.1;
(b) Si mostri che esso definisce un procedimento effettivo.

Soluzione

Esercizi L2 - variante

Esercizio (2.bis)

Gli algoritmi forniscono procedimenti effettivi ma non eseguibili per le funzioni calcolabili. Questi procedimenti sono utilizzati, come un canovaccio, per la loro implementazione in programmi che a loro volta, forniscono processi automatici e dunque, eseguibili. Si commentino gli altri 2 principali usi, visti nel corso, di tali procedimenti.

Soluzione

Esercizi L2 - 28/2/2018

Esercizio (3)

La funzione π_1 sotto, è calcolabile anche se non sapremo mai, forse, trovare un procedimento effettivo per calcolarla.

$$\pi_1(n) = \begin{cases} 1 & \texttt{se l'espansione decimale di } \pi \texttt{ contiene n occorrenze di 1 in sequenza} \\ 0 & \texttt{altrimenti} \end{cases}$$

(a) Sapreste fornire una classe di programmi nei quali è certamente contenuto anche **un** *programma che calcola π_1?*
(b) Sapreste giustificare l'affermata calcolabilità di π_1?

Soluzione
(a)
Si. La classe è composta da programmi $\mathtt{p_k}$, scritti in C (o qualunque altro **LP**), calcolanti funzioni $\mathtt{g_k}$ su interi non negativi, così definite:

$$\mathtt{g_k(n)} = \begin{cases} 1 & \texttt{se n} < \texttt{k} \\ 0 & \texttt{altrimenti} \end{cases}$$

Ognuna di queste funzioni è calcolabile ed è calcolata da uno di tali programmi. All'insieme $\{\mathtt{p_k} \mid \mathtt{k} \in \mathcal{N}\}$ aggiungiamo il programma `uno` che calcola la funzione $\mathtt{g(n)} = 1$ su ogni n non negativo.
(b)
Abbiamo visto in (a), che: $\pi_1 \in \mathtt{G} \equiv \{\mathtt{g_k} \mid \mathtt{k} \in \mathcal{N}\} \cup \{\mathtt{g}\}$. Ognuna di queste funzioni è banalmente esprimibile con un programma (ed infatti, ha un andamento che la rende continua nella topologia di Dana Scott, vedi Lezione del ...) come in (a). Non sappiamo ad oggi, quale tale le funzioni in $\mathtt{G}$ sia π_1 ma sappiamo che c'è quindi sappiamo che è calcolabile.

Esercizi L2 - 28/2/2018

Esercizio (4)

L'articolo di N. Jonoska et al. sul calcolo molecolare, citato a lezione, descrive un algoritmo, basato su DNA flexible tiles, per la 3-Colorabilità su grafi planari 4-regolari. Si esamini la definizione data alle pagine 509-510 dell'articolo e si risponda alle seguenti domande:

(a) Perchè il procedimento in 4 passi/operazioni [1) anneal; 2) ligate; 3) cleave; 4) extract.] riportato in pag. 510 definisce un algoritmo?

(b) Sapreste fornire un programma in C ovvero, in altro linguaggio di programmazione, di vostra conoscenza, che implementi tale algortmo?

(c) Sapreste fornire un programma in C che introduca una rappresentazione dei flexible tiles ed emuli su tali tiles le operazioni anneal, ligate, cleave, extract?

(d) Discutere la relazione tra la soluzione data al punto (b) e quella data al punto (c).

Soluzione

Esercizi L2 - 28/2/2018

Esercizio (5)

Consideriamo la funzione π, così definita:

$$\pi(n) = \begin{cases} 1 & \texttt{se l'espansione decimale di } \pi \texttt{ contiene la sequenza delle cifre di n} \\ 0 & \texttt{altrimenti} \end{cases}$$

(a) Sapreste fornire una classe di programmi nei quali è certamente contenuto anche **un** *programma che calcola π?*
(b) Sapreste fornire argomenti per affermare, o per negare la calcolabilità di tale funzione?.

Soluzione
(a)
Ad oggi di questa funzione si sa ben poco, in particolare sul suo andamento. Taluni ipotizzano che l'espansione contenga ogni sequenza finita di cifre decimali: In questo caso la funzione calcola 1 su tutti gli interi non negativi. Al momento nessuno conosce una classe di programmi come quella richiesta (peraltro data invece, per la funzione in esercizio L2.3).
(b)
Non abbiamo argomenti per affermare che la funzione π sia calcola nè il contrario. Le approssimazioni date per l'espansione di π ci offrono una funzione π con qualche milione di punti (più o meno isolati) che hanno immagine 1 e un'infinito di punti su cui ancora nessuno ha provato a calcolare l'occorrenza della relativa sequenza: Ad esempio: 25252525228 non è stato trovato nelle prime $2 * 10^9$ cifre decimali (vedi: //www.subidiom.com/pi/pi.asp). Ma oltre?

Esercizi L2 - 28/2/2018

Esercizio (6)

SmallC è un sotto-linguaggio di C che mantiene tutti i costrutti C fatta eccezione per le procedure e costrutti correlati. Si scriva in SmallC un programma che calcoli la funzione ordinamento di sequenza. *Allo scopo si dica quale algoritmo è implementato dal programma e quali eventuali vincoli sono assunti sulla funzione calcolata.*

Soluzione

Esercizi L2 - variante

Esercizio (6.bis)

SmallC è un sotto-linguaggio di C che mantiene tutti i costrutti C fatta eccezione per le procedure e costrutti correlati. Si può scrivere in SmallC un programma che implementa l'algoritmo di Quicksort? Si giustifichi, adeguatamen- te la risposta.

Soluzione

La risposta è NO.
La ragione di questo fatto risiede sulle proprietà dell'algoritmo di Quicksort. L'algoritmo infatti, si basa sull'uso di induzione sulla dimensione della sequenza da ordinare, partizionandola in due sotto-sequenze contigue e reciprocamente ordinate, se questa ha dimensione maggiore di ... .
Nei linguaggi di programmazione l'induzione è espressa mediante un meccanismo noto come ... .
Questo meccanismo richiede l'uso di ...

Esercizi L2 - Variante

Esercizio (6.ter)

*SmallC è un sotto-linguaggio di C che mantiene tutti i costrutti C fatta eccezione per le procedure e costrutti correlati. Si può dimostrare che SmallC∈**LP**. Si dica cosa implica tale fatto?*

Soluzione

Esercizi L2 - 28/2/2018

Esercizio (7)

L'algortitmo di quicksort definito originariamente da Hoare non pone vincoli su:
i) Dimensione finita della sequenza considerata;
ii) Tipo di valori formanti la sequenza;
iii) Relazione di ordinamento applicata.
Di modo che lo stesso algoritmo è applicabile ad ogni sequenza di valori rappresentabili nel linguaggio e su cui sia possibile definire una relazione di ordinamento (anche parziale).
Quali di questi vincoli sono, invece presenti nella soluzione data all'Esercizio L1.1 di questo corso?

Esercizi L2 - 28/2/2018

Esercizio (8)

(a) Quali tra i seguenti linguaggi sono tra loro collegati: ADA, JAVA, Haskell, SQL, Miranda, Lisp, Smalltalk, ML, C++, Pascal, OCAML?
(b) Per ogni gruppo individuato in (a) si indica natura del collegamento e fonte dell'affermazione.

Soluzione

Esercizi L2 - 28/2/2018

Esercizio (9)

Nella diffusione dei linguaggi, richiamata a lezione, sono considerate 4 tipologie di applicazioni.
(a) Si elenchino.
(b) Per ciascuna tipologia si menzioni almeno un'applicazione ed un link web alla relativa descrizione.

Soluzione

Esercizi L2 - 28/2/2018

Esercizio (10)

Si consideri il programma QSort2, allegato nei listings della settimana, Listing1, proposto come "soluzione" dell'esercizio L1.1. Si rifrasino le costruzioni utilizza- te in modo da completare il programma QSort3 sotto, equivalente a QSort2.

```
int N = ...; //Parametro di programma (costante, non modificabile)
void Scambia(...){...}
void LeggiSeq(...){...}
void StampaSeq(...){...}
void QuickSort(int Left, int Right, int A[]){...}

int main(void){
    int Seq[N];
    int k;
    LeggiSeq(Seq,&k);//Lettura sequenza da ordinare
    QuickSort(Seq,k);//Ordinamento
    StampaSeq(Seq,k);//Stampa sequenza ordinata
    return(1);
}
```

Soluzione

Esercizi L2 - 28/2/2018

Esercizio (10)

...

Soluzione

```c
int N = 15; // Parametro di programma (non modificabile)
void Scambia(int *x, int *y){
    int scambia = *x;
    *x = *y;
    *y = scambia;
}
void leggiSeq(int Seq[], int *upIndex){
    int i = 0;
    while (i<N && scanf("%d",&Seq[i])==1)i++;
    *upIndex = i-1;
}
void stampaSeq(int Seq[], int upIndex){
    for (int i=0; i<upIndex; i++) printf("%d,",Seq[i]);
    printf("%d\n",Seq[upIndex]);
}
void QuickSort(int Left, int Right, int A[]){// procedura ricorsiva
    int I=Left;  //indici in [Left..Right]
    int J=Right; //indici in [Left..Right]
    int temp=A[(Left+Right)/2];
    while (I<=J) {
        while (A[I]<temp) I++;
        while (A[J]>temp) J--;
        if (I<=J){
            Scambia(&A[I],&A[J]);
            I++; J--;
        }
    }
    if (Left<J) QuickSort(Left,J,A);
    if (I<Right) QuickSort(I,Right,A);
}

int main(void){
    int Seq[N];
    int k;
    leggiSeq(Seq,&k);    //lettura sequenza
    QuickSort(0,k,Seq);  //ordinamento
    stampaSeq(Seq,k);    //stampa sequenza
    return(1);
}
```

Esercizi L2 - 28/2/2018

Esercizio (10bis)

Si confronti la soluzione data in C con un'analoga espressa in Pascal: Quale delle 2 è più espressiva e perchè?

```
program QSort3;
    const N = 15;
    type TheSeq = array [0..N-1] of Integer;
    var  MySeq: TheSeq;
         K: Integer;
         F: Text;
    procedure leggiSeq(var Seq:TheSeq; var upIndex:Integer);
        begin ... end; (* lettura da file *)
    procedure StampaSeq(var Seq:TheSeq; upIndex:Integer);
        var i:Integer;
        begin ... end;
    procedure QuickSort(Left,Right:Integer; var A:TheSeq);
        (* type range = Left..Right *)
        var i,j,temp: Integer;
        procedure Scambia(var x, y: Integer);
            var scambia: Integer;
            begin
                scambia:=x;
                x:=y;
                y:=scambia;
            end;
        begin
            i:=left;
            j:=right;
            temp:=A[(Left+Right) div 2];
            while i<=J
            do begin
                    while A[i]<temp do Inc(i);
                    while A[i]>temp do Dec(i);
                    if (i<=J) then
                        begin
                            Scambia(A[i],A[j]);
                            Inc(I); Dec(J);
                        end;
                end;
            if Left<J then QuickSort(Left,J,A);
            if I<Right then QuickSort(I,Right,A);
        end;
begin
    leggiSeq(MySeq,K);      (* lettura sequenza *)
    QuickSort(0,k,MySeq);   (* ordinamento *)
    stampaSeq(MySeq,k);     (* stampa sequenza *)
end.
```

Esercizi L2 - 28/2/2018

Esercizio (10bis)

...

Soluzione

I 2 programmi hanno struttura molto simile infatti sono scritture in linguaggi differenti di una stessa metodologia di definizione e derivazione dei programmi (nota come *programmazione strutturata* che vedremo più avanti nel corso, vedi lezioni del ...).

Tuttavia, vediamo alcune importanti differenze dovute ai differenti linguaggi usati e alla loro espressività

1) `const N 15;`
   dichiara `N` non modificabile. In C avremmo potuto usare la macro `#define N 15` che però è una direttiva al compilatore C (con l'indicazione di rimpiazzare ogni occorrenza di N, nel programma, con 15 ed impedisce di riusare il nome N nei blocchi interni del programma). In effetti, C++ ha il costrutto `const` analogo a quello Pascal.
2) `type TheSeq = ...;`
   introduce un tipo con cui definisce la rappresentazione delle sequenze (da ordinare). L'uso di tipi e la distinzione tra valori di tipo diverso che pure hanno stessa rappresentazione costituisce una differenza di espressività che si ripercuote anche sulla correttezza dei programmi ottenuti nei differenti linguaggi.
3)
```
procedure QuickSort(...);
  var ...;
  procedure Scambia(...);
  begin
    ...
  end
```
   introduce un sezionamento (noto come *fully abstract abstraction* di cui parleremo più avanti, vedi lezioni del ...) delle procedure più fine ed espressivo di quello del C. In particolare, quì è espresso che l'operazione Scambia, definita dall'omonima procedura, è utilizzata nella procedura QuickSort. Questa consapevolezza ha ricadute sulla modifica, manutenzione e correttezza del programma.

Concludiamo che i 3 punti richiamati sopra, mostrano che il programma in Pascal è più espressivo di quello in C e permette una maggiore affidabilità e correttezza del programma ed una migliore modificabilità e ri-adattamento delle varie parti del codice.

Esercizi L2 - 28/2/2018

Esercizio (11)

I programmi QSort2 e QSort3 in esercizi L1.1 e L2.10 sono equivalenti e interscambiabili dal punto di vista della funzione calcolata. Ben diverso è invece, dal punto di vista dell'espressività. Facendo riferimento a quanto visto a lezione, si dica:

(a) In cosa differiscono QSort2 e QSort3;
(b) Quale dei due è più espressivo;
(c) Quale dei due è più prossimo, e perchè, ad un'implementazione dell'algoritmo di Hoare, richiamato in L2.7

Soluzione

Esercizi L2 - 28/2/2018

Esercizio (12)

(a) Si discuta una modifica del programma incompleto di esercizio L2.10, sopra, tale che il nuovo programma possa ordinare in accordo ad un ordinamento `g` *su interi, totale ma arbitrario.*
(b) Si completi il programma ottenuto in (a) in modo da ottenere un programma QSort4 equivalente a QSort2 e a QSort3. Si evidenzi definizione e uso di `g` *nel nuovo programma.*
(c) Si compili QSort4 e si esegua il sorgente con sequenza di imput `3,-7,5,-3,7`*, opportunamente codificata in accordo alle assunzioni fatte sul formato di I/O del programma.*
(d) Si modifichi QSort4 in un programma che utilizzi un ordinamento: $\mathtt{g}(\mathtt{n},\mathtt{m})$ *sse* $|\mathtt{n}| \geq |\mathtt{m}|$*. In particolare, si mostri la nuova definizione di* `g`*, si compili ed esegua il nuovo programma con la sequenza introdotta in (c).*

Soluzione

(a)

In C possiamo aggiungere alla lista delle procedure `Scambia, LeggiSeq, StampaSeq, QuickSort` la procedura `g`. La nuova procedura dovrà calcolare un valore booleano ed essere utilizzata nel corpo della procedura QuickSort al posto delle primitive di ordinamento < e >.

Esercizi L2 - 28/2/2018

Esercizio (13)

(a) Sapreste scrivere un comando, che chiameremo NOP, in C che aggiunto ad un programma in C non modifica la funzione calcolata da tale programma.
(b) Quale funzione calcola un programma C costituito dal solo comando NOP?
(c) Quanti programmi C calcolano una stessa funzione calcolabile? Si motivi la risposta.

Soluzione
(a)
NOP: ; — comando vuoto
NOP: if(1); — comando condizionale falso (o vero, ma vuoto)
NOP: while(0); — comando iterazione vuota
e molti altri costrutti possibili.

(b)
Una funzione invariante (che non modifica lo stato o il valore corrente), ad esempio l'identità su $\mathcal{D}$.

(c)
Aleph0 perchè l'insieme di tutti i programmi ha cardinalità del numerabile e dato un programma $\mathtt{p}$, sia $\mathtt{p}^{\mathtt{n}}$ il programma $\mathtt{p}$ opportunamente, esteso con una sequenza di $\mathtt{n}$ occorrenze NOP, per ogni naturale $\mathtt{n}$. La funzione calcolata da ciascun $\mathtt{p}' \in \mathtt{P}_{\mathtt{p}} \equiv \{\mathtt{p}^{\mathtt{n}} \mid \mathtt{n} \in \mathcal{N}\}$ è la funzione calcolata da $\mathtt{p}$, e $\mathtt{P}_{\mathtt{p}}$ ha cardinalità del numerabile (vedi programma NOP.c nei listing di C/C++)