DESCRIZIONI RIGOROSE c7

Es. cucinare la pasta asciutta per una famiglia

La descrizione rigorosa della strategia risolutiva prevede

- Analisi del problema (con chiara individuazione degli obiettivi e dei dati iniziali)
- Progettazione (specificazione della sequenza di azioni da compiere)

I dati iniziali sono incompleti (qual è il il N di persone della famiglia? quanta pasta cucinare per ogni persona?)

Anche i risultati finali devono essere precisati meglio (che significa cucinare? Quale deve essere il grado di cottura? Quanto deve essere gustosa/salata la pasta? Bisogna specificare il tempo di cottura, la quantità di sale per ogni litro d'acqua).

Tutti questi elementi concorrono a stabilire i dati iniziali. Spesso i dati iniziali non sono immediatamente chiari e si arriva a determinarli durante la stesura della strategia risolutiva.

La strategia risolutiva deve essere

- Rigorosa, non ambigua (dimensioni della pentola, la quantità di acqua in pentola per ogni persona)
- Completa, contenere tutti i dati (N di persone, quantità di pasta, quantità di sale)
- Dettagliata (che significa acqua che bolle? temperatura)

Solo in questo modo l'esecutore può compiere le azioni necessarie a risolvere il problema.

Il problema è ben formulato:

- Sono presenti tutti i dati iniziali indispensabili

ed è esplicito

- L'obiettivo finale
- Il criterio di verifica

La strategia risolutiva è generale perché risolve problemi che appartengono alla stessa classe.

Es risolvere un'equazione di 2° grado: risolvere una qualunque eq di quel tipo e non solo una eq specifica.

L’ALGORITMO c8

La strategia risolutiva è caratterizzata da una serie di passi o, meglio, da una sequenza di azioni.

Azione è un evento nel quale un soggetto (esecutore) agisce su degli oggetti operando su questi una trasformazione.

Azione elementare o istruzione non può essere scomposta in azioni più semplici: è interpretabile in modo univoco e direttamente eseguibile dall’esecutore, cioè l’esecutore sa cosa deve fare e come farlo.

Le azioni devono essere eseguite in sequenza, una dopo l’altra.

La strategia risolutiva è descritta da una sequenza ordinata di azioni elementari.

ALGORITMO: è un procedimento risolutivo (che risolve un problema) costituito da un insieme finito di azioni elementari univocamente interpretabili ed eseguibili; cioè

- è un procedimento costituito da una successione ordinata e finita di istruzioni (eseguite in un certo ordine e che ad un certo punto finisce)
- parte dai valori assunti dai dati iniziali e opera su questi una trasformazione
- l’esecuzione del procedimento fornisce i risultati del problema

Caratteristiche di un algoritmo. Deve essere:

- Finito (costituito da un N finito di azioni elementari; è previsto un inizio e una fine)
- Univoco (non ambiguo, preciso, rigoroso): non deve possedere ambiguità per l’esecutore
- Generale (valido per tutti i problemi della stessa classe)
- Completo (deve considerare tutti i casi possibili e l’azione da eseguire in ogni caso)
- Osservabile nei risultati (deve arrivare ad un risultato oggettivo)
- Deterministico (per ogni unità di tempo l’esecutore deve compiere una sola azione e partendo dagli stessi dati iniziali deve arrivare sempre alla stessa soluzione)

Ad ogni strategia risolutiva corrisponde un algoritmo (sequenza di azioni).

EFFICIENZA: parametro di bontà di un algoritmo (consente di stabilire quale strategia risolutiva, quale algoritmo, è migliore).

Un algoritmo è efficiente quando:

- È corretto (produce il risultato voluto)
- È veloce (in termini di tempo impiegato per produrre il risultato)
- È parsimonioso (in termini di risorse impegnate per produrre il risultato)

Un algoritmo completo deve prevedere (mediante diversi percorsi logici) le azioni da svolgere al verificarsi di determinate situazioni (Es: bancomat fuori uso, codice segreto errato).

RAPPRESENTAZIONE DEGLI ALGORITMI c9

Un algoritmo vene rappresentato mediante

- Diagrammi a blocchi (DaB e flow-chart)
- Pseudolinguaggio (linguaggio di progetto)

I diagrammi a blocchi sono una descrizione grafica dell'algoritmo, con appositi simboli, che evidenziano il flusso di esecuzione delle istruzioni

INIZIO algoritmo FINE algoritmo UNA AZIONE o UN BLOCCO DI AZIONE ORDINE DI ESECUZIONE CONDIZIONE INPUT (per fornire informazioni al computer durante l'esecuzione OUPUT per ricevere informazioni dal computer durante l'esecuzione

Il linguaggio naturale: è utilizzato tra risolutore ed esecutore umani (e utilizzato per indicare le azioni negli algoritmi analizzati fino ad ora).

L'inconveniente del linguaggio naturale: è ambiguo (può usare termini sinonimi, metafore, allegorie…)

Lo PSEUDOLINGUAGGIO è un linguaggio formale: usa simboli con un significato univoco.

La descrizione formale dell'algoritmo in pseudolinguaggio si chiama pseudocodice.

La scrittura in pseudocodice si chiama pseudocodifica.

Con la pseudocodifica il risolutore esprime le istruzioni, usando frasi ed espressioni elementari in forma naturale, concentrandosi sulla risoluzione logica del problema, senza badare alla forma.

Ogni programmatore usa un suo pseudolinguaggio usando parole chiave che rendono intuitiva la sua interpretazione.

REGOLE DELLO PSEUDOLINGUAGGIO

- Parole chiave (parole riservate scritte in maiuscolo, verbi usati all'imperativo: SCRIVI, LEGGI…)
- Altre risorse (variabili con una sola lettera in maiuscolo: A, K…; variabili a più caratteri con iniziale in maiuscolo e senza spazi: Somma, Totale, Media, NumeroMax…; costanti interamente in maiuscolo e con più di una lettera: PIGRECO, MASSIMO, MINIMO…)
- Sintassi di una istruzione
    - Parole tra parentesi angolari < > rappresentano le categorie sintattiche, cioè elementi generali del linguaggio che devono essere specificati ulteriormente. Ad es:
    - Parentesi quadre [ ] indicano opzionalità
    - Blocchi separati con il simbolo| possono essere usati in alternativa
    - Le parentesi graffe { } indicano la ripetizione
    - Il simbolo // è utilizzato per inserire commenti alle istruzioni, ignorati dall'esecutore.

Lo pseudolinguaggio è usato per esprimere in modo chiaro e semplice la soluzione logica del problema, non può essere eseguito dal calcolatore, ma deve essere tradotto in un linguaggio di programmazione che possa essere interpretato dal computer (es: linguaggio C, linguaggio Java…).

VARIABILI E COSTANTI c10

Una VARIABILE è un'area della memoria RAM riservata a contenere un particolare dato soggetto a modifiche durante l'elaborazione.

La cella di memoria contiene in ogni istante il valore della variabile.

La variabile è caratterizzata da

- Un nome che identifica la cella di memoria (ad es Base).
  Conviene assegnare alla variabile un nome che richiama il suo significato
- Un valore (dato numerico o di altro tipo)
- Un tipo (ad es intero, reale, alfanumerico, booleano)

Ad es (Base, 10, Intero)

Dichiarazione (o tipizzazione) è l'operazione con la quale si definisce qual è il tipo della variabile.

Una COSTANTE rappresenta una cella di memoria che contiene un dato statico, predefinito, che non è soggetto a modifiche durante l'elaborazione.

La costante è caratterizzata da

- Un identificatore
- Un valore

Ad es (PIGRECO, 3,14)

Variabili e costanti sono normalmente rappresentate da scatole che simboleggiano le celle di memoria riservate a contenere determinati valori durante l'elaborazione.

CLASSIFICAZIONE dei dati (sulla base dell'interazione con il computer)

- Dati di INPUT forniti dall'esterno
- Dati di OUTPUT comunicati all'esterno (soluzione del problema)
- Dati di LAVORO utilizzati durante l'elaborazione (intermedi)

CLASSIFICAZIONE dei dati (sulla base del tipo)

- Numerici numeri (interi o reali) utilizzati nelle espressioni matematiche (10, 3,14)
- Alfanumerici (o stringhe) caratteri (cifre, lettere dell'alfabeto, segni di interpunzione, car. speciali) non possono essere utilizzati per fare calcoli (un nome, un numero telefonico)

CONVENZIONI di scrittura in pseudolinguaggio

- Variabili
  - in maiuscolo se composte da una sola lettera (A, K)
  - con iniziale maiuscola e senza spazi se composta da più lettere (Somma, NumeroMax)
- Costanti
  - Nomi di più lettere completamente in maiuscolo (PIGRECO, MASSIMO, MINIMO)

TIPI DI DATI: IL TIPO INTERO c11

Un tipo di dato è caratterizzato da

- Un dominio: insieme di valori possibili che il calcolatore può rappresentare
- Un insieme di operazioni che si possono applicare ai dati (possono essere aritmetiche o relazionali)

Ogni tipo di dato ha una diversa occupazione di memoria (un numero intero occupa meno memoria di un numero in doppia precisione).

CLASSIFICAZIONE dei tipi di dati

- Tipi elementari (o tipi semplici) interi, reali, carattere, booleani
- Tipi strutturati aggregazione di altri dati semplici o strutturati dai quali è possibile estrarre i dati con operazioni appropriate

In ogni linguaggio di programmazione esistono dati predefiniti e altri possono essere definiti se necessario (tipi di dati astratti o ADT)

IL TIPO INTERO (è un tipo base)

Generalmente un intero occupa 2 o 4 byte di memoria (16 o 32 bit): il bit più significativo indica il segno (0 positivo, 1 negativo), gli altri sono riservati al numero. Se N è il numero di bit a disposizione, per rappresentare un tipo intero si utilizza la rappresentazione in modulo e segno per i numeri positivi e in complemento a 2 per quelli negativi.

Il dominio è un sottoinsieme dei numeri interi relativi Z.

Gli elementi di questo sottoinsieme sono valori costanti compresi in un intervallo che dipende dal numero N di bit utilizzati per rappresentare il n° intero [da $-2^{(N-1)}$ a $+2^{(N-1)}-1$]; con N=16 l'intervallo di valori va da (-32768 a +32767).

Gli operatori che agiscono su operandi interi possono essere aritmetici (+, -, *, DIV, MOD) e il risultato è ancora un intero, oppure relazionali (<, >, ≤, ≥, ≠) e il risultato è booleano.

Nel calcolo il computer segue le regole dell'algebra rispettando priorità e precedenze.

Errori di run-time (errori a tempo di esecuzione): errori segnalati in fase di esecuzione del programma

- errore di overflow (il risultato supera l'estremo superiore)
- errore di underflow (il risultato è minore dell'estremo inferiore)
- errore di divisione per 0
- errore di modulo negativo

TIPI DI DATI: REALE, CARATTERE, STRINGA, BOOLEANO c12

IL TIPO REALE (è un tipo base)

I valori di tipo reale sono rappresentati in virgola mobile, cioè con una coppia di numeri (m, e) detti mantissa ed esponente.

Il dominio è un sottoinsieme dei numeri interi reali R.

I suoi elementi sono valori costanti compresi in un intervallo che dipende dal numero N di bit utilizzati per la mantissa e per l'esponente.

I numeri reali possono essere rappresentati in precisione singola (32 bit: 1 di segno, 8 per l'esponente e 23 per la mantissa) o in precisione doppia (64 bit: 1 di segno, 11 per l'esponente e 52 per la mantissa).

Gli operatori aritmetici sono (+, -, *, /) e il risultato è ancora un reale, gli operatori relazionali sono i soliti (<, >, ≤, ≥, ≠) e il risultato è booleano.

IL TIPO CARATTERE E IL TIPO STRINGA

Il dominio del tipo carattere è l'insieme di tutti i caratteri disponibili al computer (lettera dell'alfabeto, segno di operazione, di interpunzione, lo spazio blank e altri segni non disponibili nella tastiera, ma rappresentabili attraverso una combinazione di tasti (Alt + n° in codice ASCII per sistemi con tale codice).

Mediante gli operatori di tipo relazionale è possibile effettuare confronti sui caratteri: (0<9<A<Z<a<z).

I dati carattere sono racchiusi tra apici (carattere '2' ≠ numero 2).

Una stringa è una sequenza di caratteri ed è racchiusa tra virgolette ("Ciao, come stai?").

IL TIPO BOOLEANO

Il dominio del tipo booleano è l'insieme costituito dai 2 valori di verità (costanti logiche Vero e Falso) con Falso < Vero.

Gli operatori booleani sono, in ordine di precedenza, NOT, AND, OR.

TIPI ORDINATI i tipi intero, reale e carattere sono tipi ordinati, cioè, ad essi è possibile applicare gli operatori di relazione.

ESPRESSIONI E LORO VALUTAZIONE c13

VALUTAZIONE DELLE ESPRESSIONI

Per calcolare il valore di un'espressione che contiene più operatori, l'esecutore deve

- PRIMA stabilire l'ordine di esecuzione dei vari operatori e
- POI eseguire i calcoli veri e propri

Processo di valutazione è l'attività svolta dall'esecutore (prima di eseguire le operazioni) e consiste nel stabilire in quale ordine gli operatori presenti devono essere applicati agli operandi dell'espressione. L'esecutore prende il nome di valutatore.

Precedenza degli operatori. Ogni operatore ha la sua priorità (precedenza rispetto ad altri operatori).

* e / hanno la stessa priorità, + e / hanno differenti priorità, con l'uso delle parentesi si forza l'esecutore a modificare l'ordine di esecuzione.

AMBIENTE DI VALUTAZIONE DELLE ESPRESSIONI

Data una generica espressione, per calcolarne il risultato, il valutatore deve tener conto del suo ambiente di valutazione e della priorità e proprietà degli operatori.

L'ambiente di valutazione è l'insieme delle terne utilizzato per valutare un'espressione.

{(NomeVariabile1, Valore1, Tipo1), (NomeVariabile2, Valore2, Tipo2), …}

LE ISTRUZIONI OPERATIVE c14

TIPI DI ISTRUZIONI

Classificazione delle istruzioni di un algoritmo in base al loro comportamento

- Istruzioni operative
- Istruzioni di controllo

Le istruzioni operative corrispondono ad azioni direttamente eseguibili dall'esecutore:

- Istruzioni di assegnazione
- Istruzioni di input
- Istruzioni di output

Le istruzioni di controllo consentono di scegliere percorsi differenti durante l'esecuzione:

- Istruzioni di sequenza
- Istruzioni di selezione
- Istruzioni di iterazione

ISTRUZIONI DI INIZIO E FINE

In Pseudolinguaggio

ALGORITMO <NomeAlgoritmo>
INIZIO

FINE

DICHIARAZIONE DEGLI IDENTIFICATORI

Gli identificatori di costanti e variabili, prima di essere utilizzati, devono essere dichiarati specificandone il tipo con le parole chiave COSTANTI e VARIABILI e il simbolo ← (solo in pseudolinguaggio)

COSTANTI <Costante> ← <Valore> **VARIABILI** <Variabile> {,<Variabile>} : <Tipo>

ISTRUZIONE DI ASSEGNAZIONE

Attribuisce ad una variabile il valore di una costante, una variabile o una espressione posta a destra del simbolo ←.

In Pseudolinguaggio

< Variabile > ← <Espressione>

Inizializzazione: assegnazione del primo valore a una variabile non ancora usata

ISTRUZIONE OPERATIVA DI INPUT

Consente di assegnare ad una variabile un valore fornito dall'esterno

In Pseudolinguaggio

LEGGI(<Variabile> {,<Variabile>})

ISTRUZIONE OPERATIVA DI OUTPUT

Consente di visualizzare il valore di una variabile o di un'espressione, o messaggi sul video o sulla stampante

In Pseudolinguaggio

SCRIVI([<Messaggio>], [<Variabile >])