Informatica B - Esercitazione 1

Algoritmi e schemi a blocchi

Stefano Cereda
stefano.cereda@polimi.it

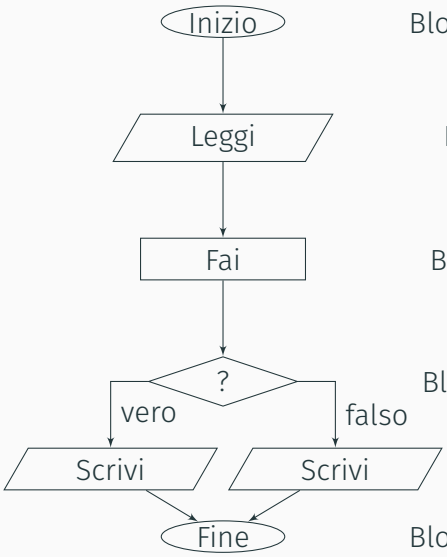
24/09/2019

Politecnico Milano

Approccio ai problemi

1. Comprensione problema
2. Divide et impera
3. Raffinamenti successivi
4. Scrittura soluzione
5. Test della soluzione

Schemi a blocchi — Costrutti



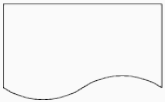
Blocco di inizio

Blocco di ingresso dati

Blocco di esecuzione

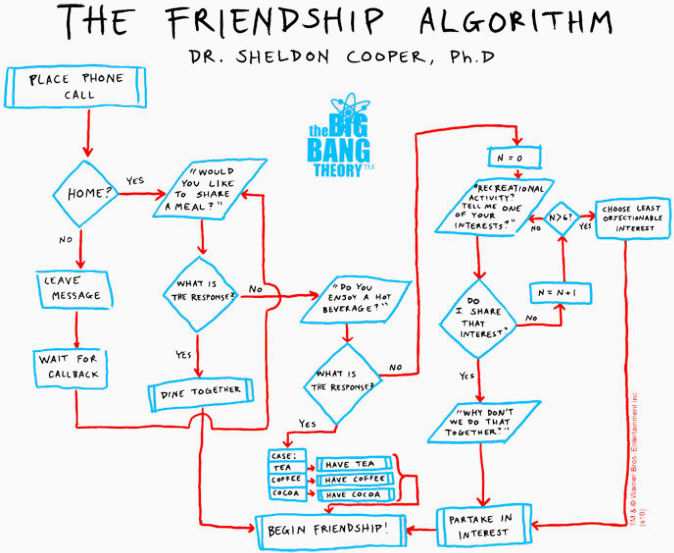
Blocco di condizione (selezione/iterazione)

Blocco di uscita dati



Blocco di fine

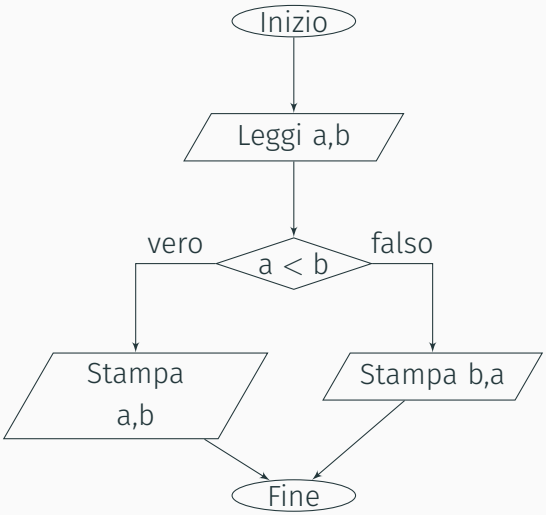
Esempio schema a blocchi



Esempio — Ordinamento coppie di numeri

Scrivere un programma che, ricevuti in input due numeri a e b, li stampi in ordine dal più piccolo al più grande.

Esempio — Soluzione con schema a blocchi



Esempio — Soluzione in pseudocodice

Pseudocodice

1. Leggo due numeri *a* e *b*
2. Se *a* è minore di *b*:
   2.1 Stampa su schermo *a* e *b*
3. Altrimenti:
   3.1 Stampa su schermo *b* e *a*
4. Fine

Linguaggio C

```c
#include <stdio.h>

void main(){
    int a,b;

    scanf("%d", &a);
    scanf("%d", &b);

    if (a < b){
        printf("%d %d\n", a, b);
    }
    else{
        printf("%d %d\n", b, a);
    }
}
```

Per gli esercizi successivi trovate le soluzioni in C su beep.

Verifica anno bisestile

Scrivere un programma che, ricevuto in ingresso un anno n, dica se è un anno bisestile.

Un anno è bisestile quando è multiplo di 4. Se è multiplo di 100 non è bisestile, a meno che sia multiplo di 400.

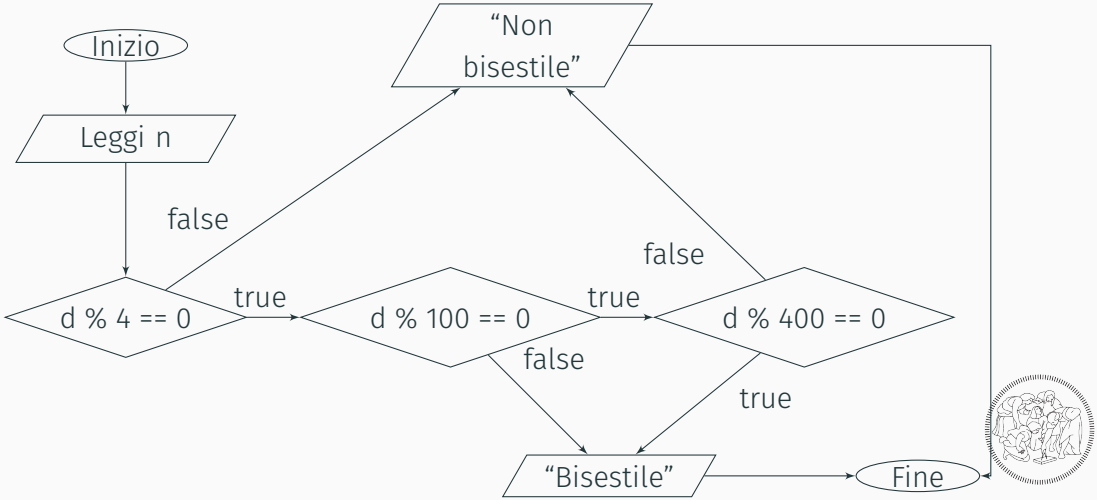
Anno bisestile — Soluzione in pseudocodice

1. Leggi n
2. Se n non è divisibile per 4 l'anno non è bisestile ed esci
3. Se n non è divisibile per 100 l'anno è bisestile ed esci
4. Se n è divisibile per 400 l'anno è bisestile ed esci
5. L'anno non è bisestile ed esci

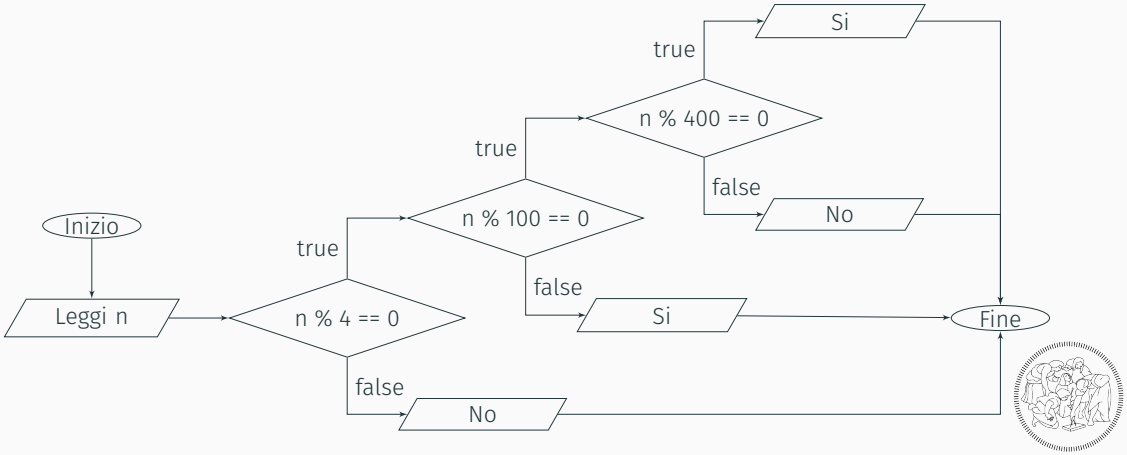
Anno bisestile — Soluzione in pseudocodice migliore

1. Leggi n
2. Se n è divisibile per 4
   - 2.1 Se n è divisibile per 100
     - 2.1.1 Se n è divisibile per 400 allora è bisestile (4 si, 100 si, 400 si), altrimenti non è bisestile (4 si, 100 si, 400 no)
   - 2.2 Altrimenti è bisestile (4 si, 100 no)
3. Altrimenti non è bisestile (4 no)
4. Fine

Anno bisestile — Soluzione con schema a blocchi



Anno bisestile — Soluzione con schema a blocchi migliore



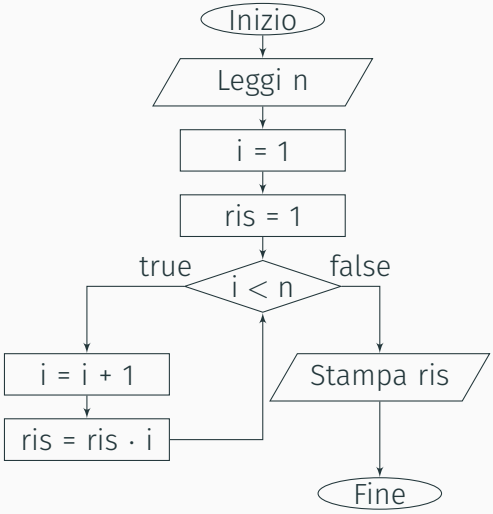
Calcolo del fattoriale

Scrivere un programma che, ricevuto in ingresso un numero n, ne calcoli e visualizzi il fattoriale.

Si assuma che l'utente non inserisca mai numeri minori di 1.

Si ricorda che il fattoriale di n è definito come il prodotto di tutti i numeri fra n ed 1: $n! = \prod_{i=1}^{n} i = 1 \cdot 2 \ldots (n-1) \cdot n$

Calcolo del fattoriale — Soluzione con schema a blocchi



Il blocco di condizione viene utilizzato per iterare delle istruzioni.

Calcolo del fattoriale — Soluzione in pseudocodice

1. Leggi n
2. Assegna 1 alla variabile i
3. Assegna 1 alla variabile ris
4. Ripeti finché $i < n$:
   4.1 Incrementa i di 1
   4.2 Assegna a ris il risultato di $ris \cdot i$
5. Stampa il valore di ris
6. Fine

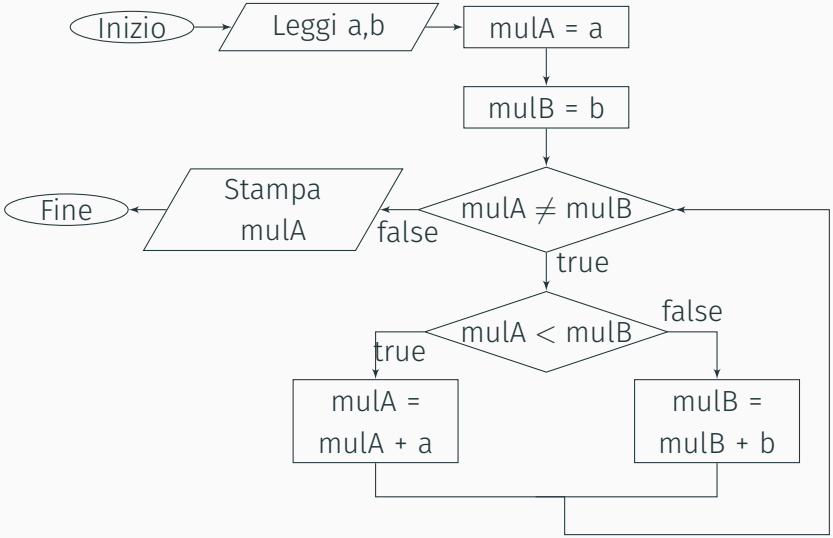
Calcolo del minimo comune multiplo

Scrivere un programma che, ricevuti in ingresso due numeri a e b, ne calcoli e visualizzi il minimo comune multiplo.

Si assuma che a e b siano sempre positivi.

Si ricorda che il minimo comune multiplo è definito come il più piccolo intero positivo multiplo sia di a che di b.

Calcolo del minimo comune multiplo — Soluzione con schema a blocchi



Calcolo del minimo comune multiplo — Soluzione in pseudocodice

1. Leggi a e b
2. Inizializza $mulA$ e $mulB$ rispettivamente con a e b
3. Ripeti finché $mulA! = mulB$:
   3.1 Incrementa il minore fra $mulA$ e $mulB$ del rispettivo valore (a o b)
4. Stampa $mulA, mulB$
5. Fine

Verifica numero primo

Scrivere un programma che, ricevuto in ingresso un numero n, dica se questo è un numero primo oppure no.

Si assuma che n sia sempre un numero positivo maggiore di 1.

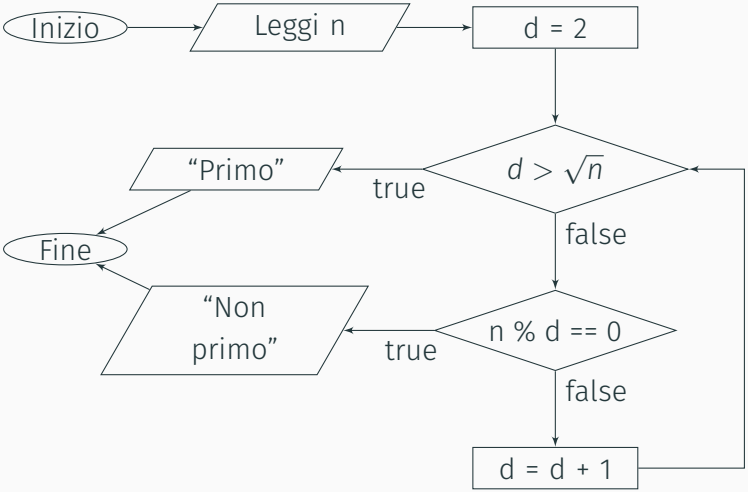
Verifica numero primo — Soluzione in pseudocodice

Cerchiamo un divisore di n:

1. Leggi n
2. Supponi che n sia primo
3. Inizializza d a 2
4. Ripeti finché $d < n$:
   4.1 Se d è divisore di n il numero non è primo
   4.2 Incrementa d di 1
5. Fine

Tip: non è necessario controllare fino ad n

Verifica numero primo — Soluzione con schema a blocchi



Si consideri il seguente frammento di programma scritto in un linguaggio macchina. All'interno di ciascuna cella di memoria c'è un'istruzione il cui significato è riportato a destra. L'indirizzo di ciascuna cella è riportato a sinistra.

Si risponda alle seguenti domande:

0000	0100000000001001	Acquisisci il numero e conservalo in X
0001	0100000000001010	Acquisisci il numero e conservalo in Y
0010	0000000000001010	Carica nel registro A il numero conservato in Y
0011	1111000000000000	Se registro A == 0 salta a 0000
0100	0000000000001001	Carica nel registro A il numero conservato in X
0101	1111000000000000	Se registro A == 0 salta a 0000
0110	0101000000001001	Stampa il contenuto di X
0111	0101000000001010	Stampa il contenuto di Y
1000	1101000000000000	Termina l' esecuzione
1001		X
1010		Y
1011		

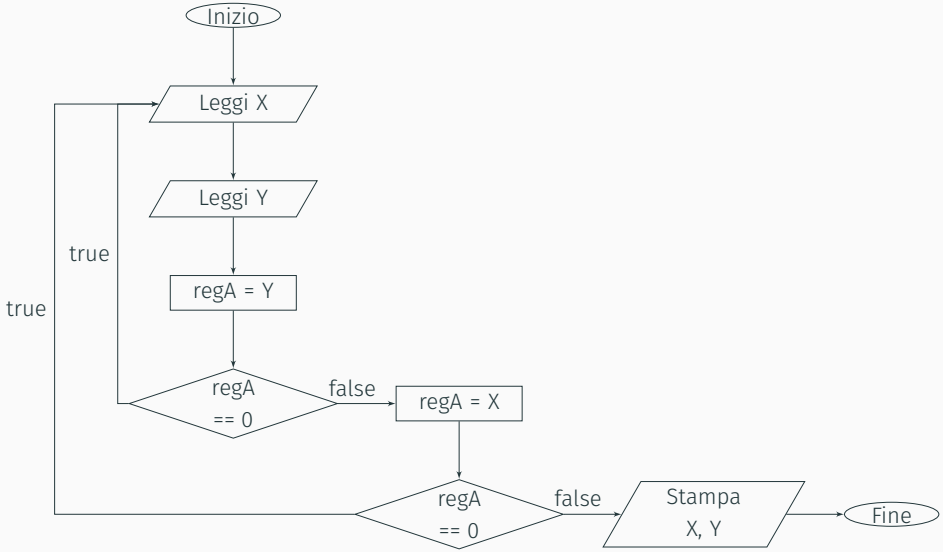
1. Si descriva il comportamento dell'unità di elaborazione durante l'esecuzione dell'istruzione che si trova nella cella di indirizzo 0101.
2. Si costruisca lo schema a blocchi (anche chiamato diagramma di flusso) corrispondente al programma riportato sopra.
3. Si scriva un programma in linguaggio C che sia equivalente a quello dato.

All'inizio dell'esecuzione dell'istruzione, il PC contiene il valore 0101.

Nella fase di fetch l'unità di elaborazione acquisisce dalla memoria l'istruzione che si trova in questa cella di memoria e la copia nel Current Instruction Register (CIR). Il PC assume valore 0110.

Nella fase di interpretazione l'unità di controllo riconosce l'istruzione.

Nella fase di esecuzione l'unità di elaborazione verifica se il valore contenuto nel registro A è pari a 0. In caso affermativo, il valore del PC viene posto al valore dell'operando dell'istruzione e diventa quindi pari a 0000.


Inizio
Leggi X
Leggi Y
regA = Y
regA
== 0
false
regA = X
regA
== 0
false
Stampa
X, Y
Fine
true
true

Ricerca in sequenza ordinata

Scrivere lo pseudocodice per cercare un valore all'interno di una sequenza, sfruttando il fatto che la sequenza è ordinata.

Ricerca in sequenza ordinata — Soluzione in pseudocodice I

1. Ricevo una sequenza s di lunghezza l e l'elemento cercato c
2. Se $l == 1$ controllo l'unico elemento della lista e:
   - 2.1 Se $e == c$ allora ho trovato c e termino
   - 2.2 Se $e \neq c$ allora c non esiste in s e termino
3. Altrimenti:
   - 3.1 Considero l'elemento e a metà di s
   - 3.2 Se $e > c$ vado a 2 utilizzando la prima metà di s come nuova sequenza
   - 3.3 Altrimenti torno a 2 utilizzando la seconda metà di s (incluso e) come nuova sequenza

Posso aggiungere una condizione 3.4 che controlla direttamente se $e == c$.

Se la stringa è di lunghezza pari posso prendere l'elemento immediatamente successivo (o precedente) alla metà esatta.

Altri Esercizi

Provate a risolvere con schemi a blocchi, pseudocodice e C i seguenti esercizi.

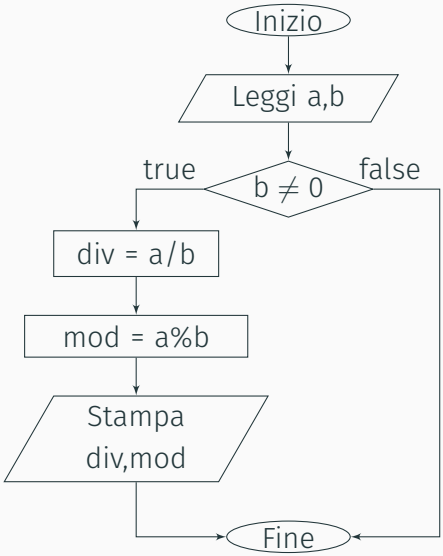
Calcolo della divisione e del resto

Scrivere un programma che, ricevuti in ingresso due numeri a e b, stampi il risultato della divisione a/b ed il relativo resto.

Se l'operazione non è possibile, il programma termina senza stampare alcun messaggio.

Utilizzare l'operatore modulo % per calcolare il resto della divisione.

Calcolo della divisione e del resto — Soluzione con schema a blocchi



Calcolo della divisione e del resto — Soluzione in pseudocodice

1. Leggo due numeri *a* e *b*
2. Se *b* è uguale a zero termino
3. Assegno a *div* il risultato della divisione intera tra *a* e *b*
4. Assegno a *mod* il resto della divisione intera tra *a* e *b*
5. Stampo su schermo *div* e *mod*
6. Fine

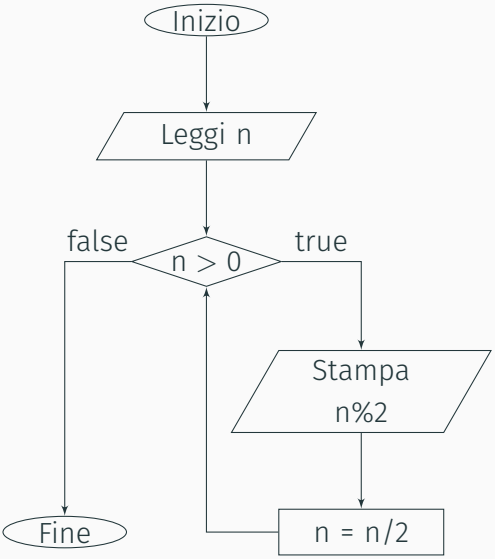
Conversione in binario

Da fare dopo aver visto il binario.

Scrivere un programma che, ricevuto in ingresso un numero positivo n, lo stampi in binario. Si utilizzi il metodo delle divisioni ripetute.

Per semplificare, si stampi separatamente ogni cifra del resto. Si stampi inoltre il risultato in ordine inverso.

Conversione in binario — Soluzione con schema a blocchi



Conversione in binario — Soluzione in pseudocodice

1. Leggi *n*
2. Ripeti finché n è maggiore di 0:
   2.1 Stampa il resto della divisione n/2
   2.2 Assegna ad *n* la parte intera della divisione n/2
3. Fine

Moltiplicazione per somme ripetute con numeri negativi

Scrivere lo pseudocodice per eseguire la moltiplicazione di numeri negativi tramite somme ripetute.

Somme ripetute — Soluzione in pseudocodice

1. Leggi $op1$ e $op2$
2. Se $abs(op1) \leq abs(op2)$:
   - 2.1 $num = abs(op2), min = abs(op1)$
   - 2.2 Altrimenti: $num = abs(op1), min = abs(op2)$
3. Se $(op1 < 0 \wedge op2 \geq 0) \vee (op1 \geq 0 \wedge op2 < 0)$:
   - 3.1 $num = -num$
4. $ris = 0$
5. Ripeti finché $min > 0$:
   - 5.1 $ris = ris + num$
   - 5.2 $min = min - 1$
6. Stampa ris e termina