INTRODUZIONE AGLI ALGORITMI

- Prima di riuscire a scrivere un programma, abbiamo bisogno di conoscere un metodo risolutivo, cioè un metodo che a partire dai dati di ingresso fornisce i risultati attesi.
- Se voglio calcolare una moltiplicazione, posso usare diversi metodi:
  - mi baso sull'addizione:

```
13 x 12 = 13 + 13 + 13 + … +  13
```

  - oppure faccio il calcolo in colonna

```
13 x
12 =
-----
 26 +
13- =
-----
156
```

1

INTRODUZIONE AGLI ALGORITMI

- Se dobbiamo spiegare ad una persona come fare una moltiplicazione, possiamo fare un esempio:

```
13 x 12 = 13 + 13 + 13 + … +  13
```

- però così abbiamo spiegato solo come si fa una particolare moltiplicazione: 13 x 12.
- Vogliamo spiegare un metodo che valga "sempre", per tutti i numeri. Dobbiamo innanzitutto definire per quali valori di ingresso funziona il nostro metodo

2

INTRODUZIONE AGLI ALGORITMI

- La persona che ci ascolta è in grado di imparare da esempi. Il calcolatore no
  - non possiamo spiegarlo con un esempio, dobbiamo dire quali passi deve svolgere
- Proviamo così:

```
A x B = A + A + A + … + A
```

Cosa vogliono dire i puntini?
Ambiguo!

3

INTRODUZIONE AGLI ALGORITMI

- La persona che ci ascolta è in grado di imparare da esempi. Il calcolatore no
  - non possiamo spiegarlo con un esempio, dobbiamo dire quali passi deve svolgere
- Proviamo così:

```
A x B = A + A + A + … +  A
```

B volte

- *Ma cosa vuol dire una somma con B termini?*
- *Es, cosa vuol dire 0 termini?*
- *Quale istruzione deve essere ripetuta B volte?*

4

INTRODUZIONE AGLI ALGORITMI

- Se vogliamo scrivere un libro di matematica di base, spiego come si fa una moltiplicazione. Devo scriverlo in modo che chiunque capisca il metodo
- non deve essere ambiguo
- dobbiamo dire quali sono i prerequisiti, le istruzioni di base che l'esecutore deve saper compiere
  - per capire questo algoritmo bisogna sapere come si calcola un'addizione
- dobbiamo dire su quali valori di ingresso si può applicare il metodo
  - sui naturali

5

SOLUZIONE: CALCOLO DEL PRODOTTO

6

IL PIU` PICCOLO NUMERO REALE >0

1. Assegna a R il valore 1
2. Dividi R per 2 e metti il risultato in R
3. Vai al passo 2
4. Stampa R

7

Calcolo π

$$\frac{\pi}{4} = \frac{1}{1} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots + \frac{(-1)^i}{2i+1} + \cdots$$

1. Assegna a *P* il valore 0
2. Assegna a *S* il valore 1
3. Assegna a *D* il valore 1
4. Calcola *S*/*D*, sommalo a *P* e metti il risultato in *P*
5. Cambia segno a *S*
6. Somma 2 a *D* e metti il risultato in *D*
7. Vai al passo 4
8. Moltiplica *P* per 4
9. Stampa *P*

8

RISOLUZIONE DI PROBLEMI

- La risoluzione di un problema è il processo che, dato un problema e individuato un opportuno metodo risolutivo, trasforma i dati iniziali nei corrispondenti risultati finali.
- Affinché la risoluzione di un problema possa essere realizzata attraverso l'uso del calcolatore, tale processo deve poter essere definito come *sequenza di azioni elementari.*

9

ALGORITMO

- *Il termine Algoritmo deriva dal nome del matematico persiano*

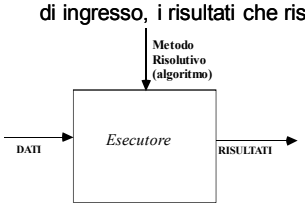
  Muhammad ibn Mūsā al-Khwārizmī

  (ca. 780-850)

10

ALGORITMO

- Un algoritmo è una sequenza **finita** di mosse che risolve ***in un tempo finito*** una *classe* di problemi.
- L'esecuzione delle azioni *nell'ordine specificato dall'algoritmo* consente di ottenere, a partire dai dati di ingresso, i risultati che risolvono il problema



ESECUTORE
una ***macchina astratta***
capace di ***eseguire le azioni***
specificate dall'algoritmo.

11

ALGORITMI: PROPRIETÀ

- **Eseguibilità**: ogni azione dev'essere *eseguibile* dall'esecutore *in un tempo finito*
- **Non-ambiguità**: ogni azione deve essere *univocamente interpretabile* dall'esecutore.

  L'italiano è ambiguo, come si vede bene dagli indovinelli

  IL MESE DI MAGGIO:

  "Ratto trascorre e a noi rose dispensa"
- **Finitezza**: il numero totale di azioni da eseguire, per ogni insieme di dati di ingresso, deve essere finito

12

ALGORITMI: PROPRIETÀ (2)

Quindi, l'algoritmo deve:

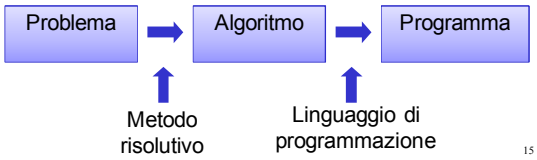
- essere *applicabile a qualsiasi insieme di dati di ingresso* appartenenti al **dominio di definizione** dell'algoritmo
- essere costituito da operazioni appartenenti ad un determinato **insieme di operazioni fondamentali**
- essere costituito da **regole non ambigue**, cioè interpretabili in modo **univoco** qualunque sia l'esecutore (persona o "macchina") che le legge

UNA VOLTA DECISO L'ALGORITMO

- Una volta che ho deciso l'algoritmo, devo fare in modo che l'elaboratore sia in grado di eseguirlo
- Le *"mosse elementari"* devono essere eseguibili dal calcolatore (quindi devo sapere quali istruzioni il calcolatore può eseguire)
- Le istruzioni vengono eseguite sui dati e forniscono dei risultati
- L'algoritmo deve essere scritto in maniera formale: codificato in un preciso linguaggio di programmazione

ALGORITMO & PROGRAMMA

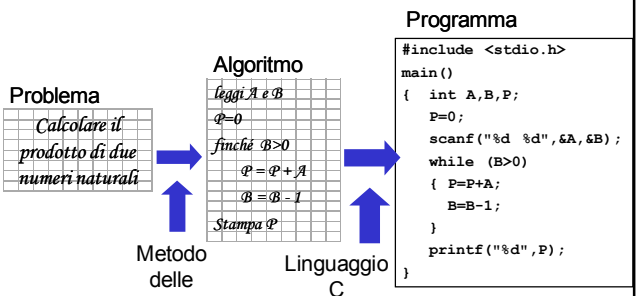
Passi per la risoluzione di un problema:
- individuazione di un procedimento risolutivo
- scomposizione del procedimento in un insieme ordinato di azioni ➡ **ALGORITMO**
- rappresentazione dei dati e dell'algoritmo attraverso un formalismo comprensibile dal calcolatore ➡ **LINGUAGGIO DI PROGRAMMAZIONE**



Metodo risolutivo

Linguaggio di programmazione

Esempio: ALGORITMO & PROGRAMMA



Metodo delle somme successive

Linguaggio C

PROGRAMMA

Un programma è un testo scritto in accordo alla **sintassi** e alla **semantica** di un linguaggio di programmazione.

Un ***programma*** è la **formulazione testuale,** in un certo linguaggio di programmazione, di un **algoritmo** che risolve un dato *problema*.

UN ESEMPIO DI PROGRAMMA (in linguaggio C)

```
main()
{
 int A, B;
 printf("Immettere due numeri: ");
 scanf("%d %d", &A, &B);
 printf("Somma: %d\n", A+B);
}
```

L'ELABORATORE ELETTRONICO

- Il calcolatore elettronico è uno strumento in grado di eseguire insiemi di *azioni* ("*mosse*") *elementari*
- le azioni vengono eseguite su oggetti (***dati***) per produrre altri oggetti (***risultati***)
- l'esecuzione di azioni viene richiesta all'elaboratore attraverso *frasi* scritte in qualche *linguaggio* (*istruzioni*)

19

PROGRAMMAZIONE

L'attività con cui si predispone l'elaboratore a **eseguire un *particolare insieme di azioni* su *particolari dati***, allo scopo di *risolvere un problema*

20

ALCUNE DOMANDE FONDAMENTALI

- **Quali istruzioni** esegue un elaboratore?
- **Quali problemi** può risolvere un elaboratore?
- ***Esistono problemi che un elaboratore non può risolvere?***
- **Che ruolo ha il linguaggio** di programmazione?

21

PROBLEMI DA RISOLVERE

- I problemi che siamo interessati a risolvere con l'elaboratore sono di natura molto varia.
  - *Dati due numeri trovare il maggiore*
  - *Dato un elenco di nomi e relativi numeri di telefono trovare il numero di telefono di una determinata persona*
  - *Dati a e b, risolvere l'equazione ax+b=0*
  - *Stabilire se una parola viene alfabeticamente prima di un'altra*
  - *Somma di due numeri interi*
  - *Ordinare una lista di elementi*
  - *Calcolare il massimo comun divisore fra due numeri dati.*
  - *Calcolare il massimo in un insieme.*

22

RISOLUZIONE DI PROBLEMI

- La descrizione del problema non fornisce (in generale) un metodo per risolverlo.
  - Affinché un problema sia risolvibile è però necessario che la sua definizione sia chiara e completa
- Non tutti i problemi sono risolvibili attraverso l'uso del calcolatore. Esistono classi di problemi per le quali la soluzione automatica non è proponibile. Ad esempio:
  - se il problema presenta infinite soluzioni
  - per alcuni dei problemi **non è stato trovato** un metodo risolutivo
  - per alcuni problemi è stato dimostrato che **non esiste** un metodo risolutivo automatizzabile

23

RISOLUZIONE DI PROBLEMI

- Noi ci concentreremo sui problemi che, ragionevolmente, ammettono un metodo risolutivo ➡ **funzioni calcolabili**.
- Uno degli obiettivi del corso è presentare le tecnologie e le metodologie di programmazione
  - **Tecnologie**: strumenti per lo sviluppo di programmi
  - **Metodologie**: metodi per l'utilizzo corretto ed efficace delle tecnologie di programmazione

24

ALGORITMI E PROGRAMMI

- Ogni elaboratore è una macchina in grado di eseguire azioni elementari su oggetti detti **DATI**.
- L'esecuzione delle azioni è richiesta all'elaboratore tramite comandi elementari chiamati **ISTRUZIONI** espresse attraverso un opportuno formalismo: il **LINGUAGGIO di PROGRAMMAZIONE**.
- La formulazione testuale di un algoritmo in un linguaggio comprensibile a un elaboratore è detta **programma**.

25

ALGORITMI: ESEMPI

- **Soluzione dell'equazione $ax+b=0$**
  - leggi i valori di a e b
  - calcola $-b$
  - dividi quello che hai ottenuto per a e chiama x il risultato
  - stampa x

NOTA: per denotare dati nell'algoritmo si utilizzano **VARIABILI** ossia nomi simbolici

26

ALGORITMI: ESEMPI

- **Calcolo del massimo di una sequenza di numeri $a_1 \ldots a_n$**

1 livello di specifica

- Scegli il primo elemento come massimo provvisorio $max \leftarrow a_1$
- Per ogni elemento a_i dell'insieme: se $a_i > max$ eleggi a_i come nuovo massimo provvisorio: $max \leftarrow a_i$
- Il risultato è max

27

ALGORITMI: ESEMPI

- **Stabilire se una parola P viene alfabeticamente prima di una parola Q**
  - **leggi** P,Q
  - **ripeti quanto segue:**
    - **se** prima lettera di P < prima lettera di Q
    - **allora** scrivi vero
    - **altrimenti se** prima lettera P > Q
    - **allora** scrivi falso
    - **altrimenti** (le lettere sono =)
    - togli da P e Q la prima lettera
  - **fino** a quando hai trovato le prime lettere diverse.

28

ALGORITMI: ESEMPI

- **Somma degli elementi dispari di un insieme**
  - Detto INS l'insieme di elementi considero un elemento X di INS alla volta senza ripetizioni. Se X è dispari, sommo X a un valore S inizialmente posto uguale a 0. Se X è pari non compio alcuna azione.
- **Somma di due numeri X e Y**

Si supponga di avere a disposizione come mossa elementare solo l'incremento e non la somma tra interi

29

ALGORITMI: ESEMPI

- **Somma degli elementi dispari di un insieme**
  - Detto INS l'insieme di elementi considero un elemento X di INS alla volta senza ripetizioni. Se X è dispari, sommo X a un valore S inizialmente posto uguale a 0. Se X è pari non compio alcuna azione.
- **Somma di due numeri X e Y**
  - Incrementare il valore di Z, inizialmente posto uguale a X per Y volte. Ovvero:
    - poni Z = X
    - poni U = 0
    - finché U è diverso da Y
      - incrementa Z (Z:=Z+1)
      - incrementa U (U:=U+1)
    - Il risultato è Z

30

ALGORITMI EQUIVALENTI

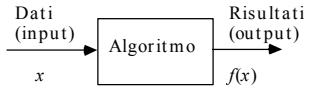
Due algoritmi si dicono ***equivalenti*** quando:

- hanno lo stesso **dominio di ingresso**;
- hanno lo stesso **dominio di uscita**;
- in corrispondenza degli **stessi valori del dominio di ingresso *producono gli stessi valori* nel dominio di uscita.**

ALGORITMI EQUIVALENTI (2)

Due algoritmi ***equivalenti***

- forniscono lo **stesso risultato**
- ma possono avere ***diversa efficienza***
- e possono essere **profondamente diversi !**



ALGORITMI EQUIVALENTI (3)

ESEMPIO: calcolo del M.C.D. fra due interi M, N

- **Algoritmo 1**
  - Calcola l'insieme A dei divisori di M
  - Calcola l'insieme B dei divisori di N
  - Calcola l'insieme C dei divisori comuni = A ∩ B
  - Il risultato è il massimo dell'insieme C
- **Algoritmo 2 (di Euclide)**

$$\text{MCD}(M,N) = \begin{cases} M \text{ (oppure } N) & \text{se } M=N \\ \text{MCD}(M-N,\ N) & \text{se } M>N \\ \text{MCD}(M,\ N-M) & \text{se } M<N \end{cases}$$

ALGORITMI EQUIVALENTI (4)

ESEMPIO: calcolo del M.C.D. fra due interi M, N

- **Algoritmo 2 (di Euclide)**
  - Finché M≠N:
    - se M>N, sostituisci a M il valore M' = M-N
    - altrimenti sostituisci a N il valore N' = N-M
  - Il Massimo Comun Divisore è il valore finale ottenuto quando M e N diventano uguali

$$\text{MCD}(M,N) = \begin{cases} M \text{ (oppure } N) & \text{se } M=N \\ \text{MCD}(M-N,\ N) & \text{se } M>N \\ \text{MCD}(M,\ N-M) & \text{se } M<N \end{cases}$$

ALGORITMI EQUIVALENTI (5)

Gli **algoritmi 1 e 2 sono equivalenti…**
…ma hanno efficienza ben diversa!!

•Es: calcolo del mcd di 324543324 e 654345432
•mcd = 12

algoritmo 1:
algoritmo 2: