



1

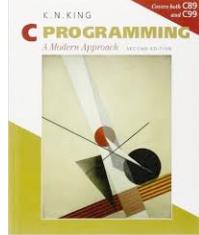


2

# RICORDATE




- Solo fino al 13 ottobre duplico, poi solo e-learning  
<http://docenti.unisa.it/020880/home>  
<http://elearning.informatica.unisa.it/el-platform/>
- Consultate @studenti.unisa.it

●

3

## AVVISI ATTIVITA'

- **Esercizi del fine settimana:** provate sempre prima a svolgerli da soli. Seguite le specifiche della traccia, i vincoli che avete...
- Non è obbligatorio inviarmi le soluzioni, ma...
- ...se avete dubbi è obbligatorio scrivermi.

per queste prime 2 settimane, non seguirò  
precisamente il testo... molte cose che vi  
racconterò (compresi gli esempi) non li trovate

●

4

## AVVISI ATTIVITA'

- **Esercizi del fine settimana:** provate sempre prima a svolgerli da soli. Seguite le specifiche della traccia, i vincoli che avete...
- Non è obbligatorio inviarmi le soluzioni, ma...
- ...se avete dubbi è obbligatorio scrivermi.
- Ho creato il Team del corso ([fate richiesta di iscrizione](#))
  - accedere a Teams
  - selezionare [Partecipa o crea un team](#)
  - selezionare [Entra in un team con un codice](#),
  - incollare il codice **6fjokp9** nella casella e selezionare [Partecipa](#).



5

## PROBLEMA

Ogni quesito  
di cui si  
richiede la  
soluzione,  
partendo di  
solito da  
elementi noti.  
(treccani)



## ALGORITMI



## PROGRAMMI

perché noi cosa vorremmo?  
Qualcuno che facesse i «conti»  
al posto nostro...



6

**Problema → Algoritmi → Programmi**

Ogni quesito di cui si richieda la soluzione, partendo da elementi noti. (treccani+CS)

Insieme finito di istruzioni finite, chiare, comprensibili e non ambigue eseguibili da un agente meccanico o umano, che consentono di risolvere il problema di partenza.

Costituito dalla rappresentazione relative ai dati e la manipolazione della rappresentazione che realizzano le funzionalità richieste (operazioni)

7

## ESEMPI SEMPLICI

**Problema: dato un intero, dimmi se è pari o dispari**

- 0) Sia X questo intero
- 1) Calcola  $X:2$  [divisione tra interi]. Sia d il risultato e N il resto [intero, tra 0,1]
- 2) Se N è zero, allora X è pari; se invece N è uno, allora X è dispari.
- E' corretto?
- E' corretto (cioè fa quello che volevo)?

**Problema: calcolare il volume di un parallelepipedo**

- 0) Sia B la base, H l'altezza, P la profondità
- 1) Il volume è il risultato di  $B \cdot H \cdot P$

già  
algoritmi

8

## PROBLEMA MOLTO FREQUENTE

Problema: consideriamo questa sequenza di 10 numeri

11, 34, 2, 56, 110, 43, 6, 10, 1, 90

ordinare dal minore al maggiore



una possibilità di soluzione...



9

## PROBLEMA MOLTO FREQUENTE

Problema: consideriamo questa sequenza di 10 numeri

11, 34, 2, 56, 110, 43, 6, 10, 1, 90

ordinare dal minore al maggiore



0) Siano  $x_1, x_2, \dots, x_{10}$  gli interi [INPUT]

1) Calcola il minimo tra  $x_1, x_2, \dots, x_{10}$ , cancellalo da questa lista e scrivilo.

2) Calcola il minimo tra  $x_1, x_2, \dots, x_{10}$  [escludendo l'elemento cancellato], cancellalo da questa lista e scrivilo a destra di quello calcolato al passo 1)

3) Calcola il minimo tra  $x_1, x_2, \dots, x_{10}$  [escludendo i 2 elementi cancellati], cancellalo da questa lista e scrivilo a destra di quello calcolato al passo 2)

4) Calcola il minimo tra  $x_1, x_2, \dots, x_{10}$  [escludendo i 3 elementi cancellati], cancellalo da questa lista e scrivilo accanto a quello calcolato al passo 3)

...

9) Calcola il minimo tra  $x_1, x_2, \dots, x_{10}$  [escludendo gli 8 elementi cancellati], cancellalo da questa lista e scrivilo accanto a quello calcolato al passo 8)

10) Scrivi l'unico elemento rimasto.

E' corretto (cioè fa quello che volevo)?

10

## ALTRO ESEMPIO

Data una sequenza di  $n+1$  interi, ognuno compreso tra 1 e  $n$ , trovare uno dei duplicati ( $n > 0$ )

**Problema**



Esiste certamente!

Input: 1,2,3,2  
Output: 2

4 numeri compresi tra 1 e 3

Input: 4,3,3,1,4  
Output: 4 o 3

5 numeri compresi tra 1 e 4



11

## ALTRO ESEMPIO

Data una sequenza di  $n+1$  interi, ognuno compreso tra 1 e  $n$ , trovare uno dei duplicati ( $n > 0$ )

**Problema**



**Forza bruta:** provo tutte le possibilità (per ogni elemento della sequenza, vedo se lo trovo successivamente...)

**Algoritmo 1**

Input: 1,2,3,2  
Output: 2

- 0) Siano  $x_1, x_2, \dots, x_{n+1}$  gli interi [INPUT]
- 1) Controlla se  $x_1$  è uguale ad un elemento tra  $x_2, \dots, x_{n+1}$ . Se lo è ho finito, l'ho trovato. Altrimenti:
- 2) Controlla se  $x_2$  è uguale ad un elemento tra  $x_3, \dots, x_{n+1}$ . Se lo è ho finito, l'ho trovato. Altrimenti:
- 3) Controlla se  $x_3$  è ...
- ...
- $n-1$ ) ... Altrimenti:
- n) **Necessariamente**  $x_n = x_{n+1}$

- È corretto (cioè fa quello che volevo)?

12

## ALTRO ESEMPIO

Data una sequenza di  $n+1$  interi, ognuno compreso tra 1 e  $n$ , trovare uno dei duplicati ( $n > 0$ )

**Problema**

**Conteggio:** creo una «struttura» che tiene conto delle ripetizioni...



**Algoritmo 2**



13

## ALTRO ESEMPIO

Data una sequenza di  $n+1$  interi, **ognuno compreso tra 1 e  $n$** , trovare uno dei duplicati ( $n > 0$ )

**Problema**

**Conteggio:** creo una «struttura» che tiene conto delle ripetizioni...



**Algoritmo 2**

Input: 1,2,3,2	1	
Output: 2	2	
	3	

meno  
operazioni  
(ma abbiamo  
bisogno di  
questa  
tabella...)

- 0) Siano  $x_1, x_2, \dots, x_{n+1}$  gli interi [INPUT].  
Crea una tabella con  $n$  righe da 1 a  $n$ .
- 1) Segna con  $\checkmark$  nella riga corrispondente a  $x_1$
- 2) Segna con  $\checkmark$  nella riga corrispondente a  $x_2$ . Se già c'è, ho finito, l'ho trovato. Altrimenti:
- 3) Segna con  $\checkmark$  nella riga corrispondente a  $x_3$ . Se già c'è, ho finito, l'ho trovato. Altrimenti:
- ...
- $n-1$ ) ... Altrimenti:
- n) **Necessariamente**  $x_n = x_{n+1}$ 
  - È' corretto (cioè fa quello che volevo)?

14

## COME SI IMPARA A SCRIVERE UN ALGORITMO?

- Non si *impara* un algoritmo, ma come usare il **ragionamento algoritmico** per **risolvere** un **problema** in modo “ottimale”
- (efficienza di calcolo)
- Esempio: **ordinare, contare ripetizioni...**  
*ci sono tanti procedimenti e strutture dati per questo*

progettazione  
sequenziale,  
top down e bottom up,  
divide et impera...  
orientata a oggetti...



1	
2	
3	

15

## COMPUTATIONAL THINKING



Computational thinking builds on the power and limits of computing processes, whether they are executed by a human or by a machine. Computational methods and models give us the courage to solve problems and design systems that no one of us would be capable of tackling alone. Computational thinking confronts the riddle of machine intelligence: What can humans do better than computers? and



Hartmanis:  
“programmatore come l’artista”

J.M. Wing, “Computational Thinking,” CACM Viewpoint, March 2006, pp. 33-35.  
<http://www.cs.cmu.edu/~wing/>

My Grand Vision for the Field...

16

# ULTIMO PASSAGGIO...

**Problema**

**Algoritmi**

**Programmi**

Dobbiamo dare un nome a:

- input
- operazioni
- output



farcì capire dal computer  
per automatizzare  
il lavoro...



17

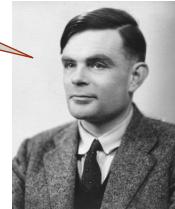
La macchina interpreta qualsiasi cosa le sia detta  
in un modo ben definito,  
senza alcun umorismo o senso delle proporzioni.

Quindi, se nel **comunicare con essa** non si dice  
**esattamente** quello che si intende,  
è destino che ne vengano guai.

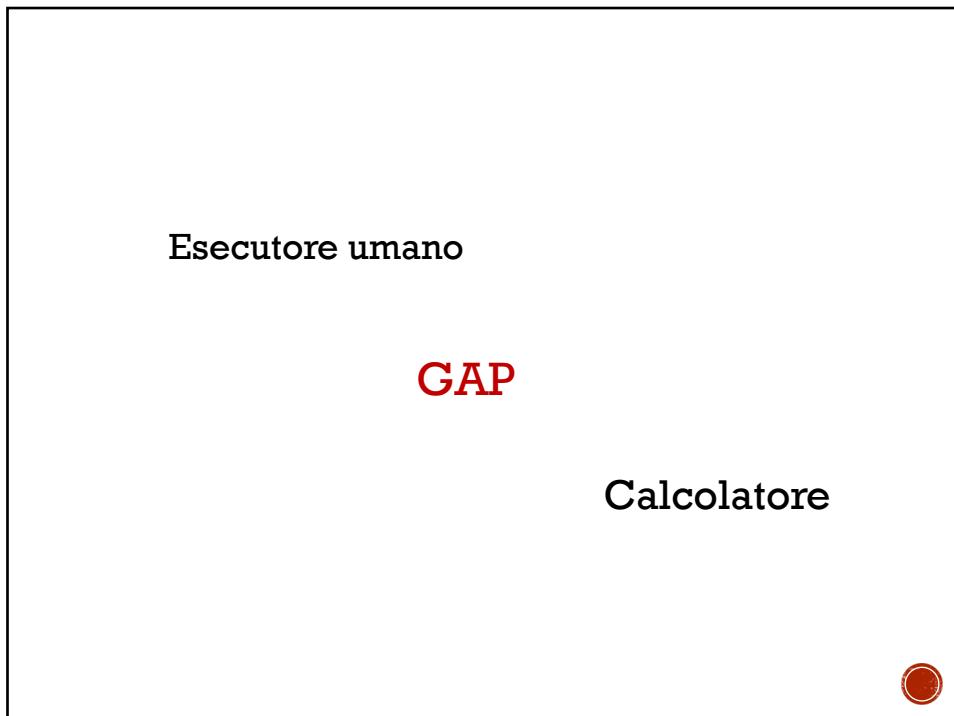
Alan M. Turing  
Intelligenza  
meccanica  
A cura di Gabriele Lelli



Serie scientifica  
Università Bocconi Bergamo



18



19

## **PROBLEMA: COME PARLO CON IL PC?**

Linguaggio di programmazione

## **Ma... cosa gli dico???**

Programma: *insieme di istruzioni espresse in un linguaggio formale (sintassi e semantica)*, il linguaggio di programmazione.

**Problema:** I computer però non sono in grado di capire nemmeno il linguaggio programmazione. Il microprocessore sa elaborare solo in linguaggio binario: ecco pertanto che i programmi dovranno essere ulteriormente tradotti da appositi applicazioni quali interpreti o compilatori.

20

# AVETE MAI VISTO... CODICE SORGENTE?

```
<!DOCTYPE html>
<html>
<head>
<title>RoundCube Webmail :: Posta in arrivo</title>
<meta http-equiv="X-UA-Compatible" content="IE=EDGE" />
<meta name="viewport" content="" id="viewport" />
<link rel="shortcut icon" href="skins/larry/images/favicon.ico"/>
<link rel="stylesheet" type="text/css" href="skins/larry/styles.css?s=1382384364" />
<link rel="stylesheet" type="text/css" href="skins/larry/mail.css?s=1382384364" />
<!--[if IE 9]><link rel="stylesheet" type="text/css" href="skins/larry/svggradients.css?s=1382384364" /><![endif]-->
<!--[if lte IE 8]><link rel="stylesheet" type="text/css" href="skins/larry/iehacks.css?s=1382384364" /><![endif]-->
<!--[if lte IE 7]><link rel="stylesheet" type="text/css" href="skins/larry/ie7hacks.css?s=1382384364" /><![endif]-->
<link rel="stylesheet" type="text/css" href="plugins/jqueryui/themes/larry/jquery-ui-1.9.1.custom.css?s=1382384363" />
<script type="text/javascript" src="skins/larry/ui.js?s=1382384364"></script>
<style type="text/css">
</style>
```



21

## INGREDIENTI DI UN LP: SINTASSI E SEMANTICA

### Sintassi

- Specifica come costruire un programma nel linguaggio di programmazione

### Semantica

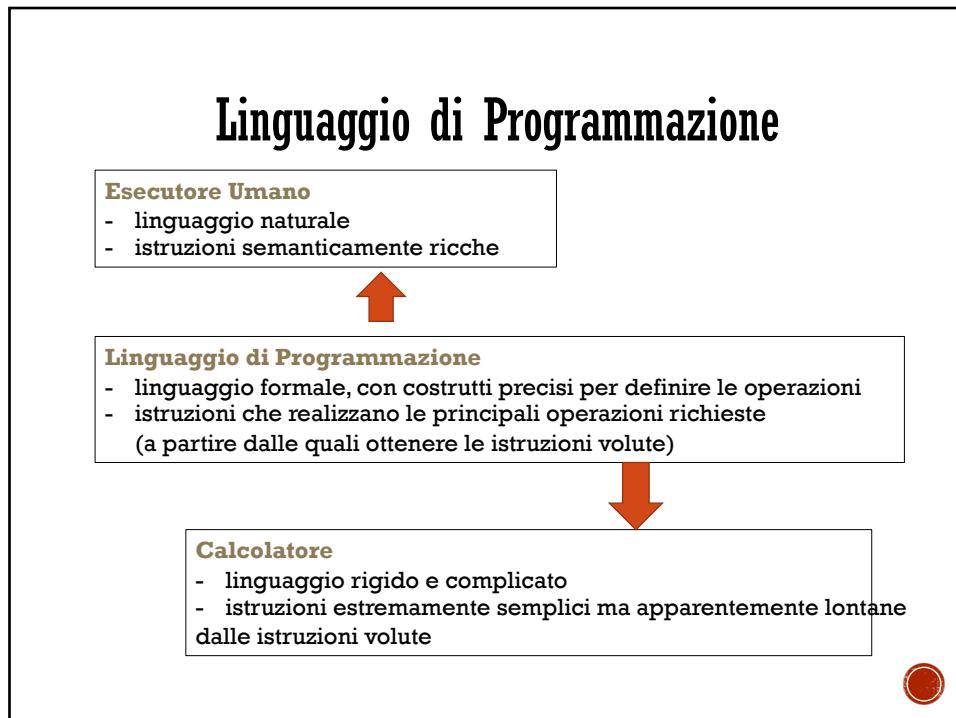
- Specifica il significato del linguaggio

#### Esempio:

- Sintassi: `<data>=CC / CC / CCCC`
- *C è un'abbreviazione per <cifra>* **01/02/2003**
- Semantica: 01/02/2003
- In Italia → 1° Febbraio 2003
- Negli USA → 2° Gennaio 2003



22



23



24

## ...UN ESEMPIO PER CAPIRE MEGLIO

supponiamo di voler prendere 200g di farina da un sacchetto utilizzando un cucchiaio.

Un linguaggio di basso livello potrebbe essere  
 1 - distendi le dita della mano  
 2 - chiudile sul pomello del cassetto delle posate  
 3 - tendi il muscolo del braccio per aprire il cassetto  
 4 - rilascia il muscolo del braccio  
 5 - distendi le dita della mano  
 6 - tendi i muscoli del braccio per portare la mano sopra i cucchiali  
 7 - chiudi le dita della mano sopra un cucchiaio  
 8 - tendi i muscoli del braccio per estrarre il cucchiaio dal cassetto  
 9 [...]

Un linguaggio un po' più astratto potrebbe essere  
 1 - prendi un cucchiaio  
 2 - prendi il sacchetto della farina  
 3 - prendi la bilancia  
 4 - prendi un contenitore  
 5 - posa il contenitore sulla bilancia  
 6 - versa tanta farina nel contenitore finché non raggiungi i 200g

Mentre un linguaggio di livello ancora più alto potrebbe fare direttamente

- prendi 200g di farina con un cucchiaio

L'astrazione si può identificare con la "distanza" dalla macchina e la "vicinanza" all'uomo.

25

## PARADIGMI (*STILI*) DI PROGRAMMAZIONE

**IMPERATIVI:**  
 l'istruzione è un comando esplicito, opera su uno o più dati oppure sullo stato interno della macchina, e le istruzioni vengono eseguite in un ordine prestabilito.  
 [C.PASCAL]

**ORIENTATA AGLI OGGETTI:**  
 programmare = descrivere l'insieme delle relazioni fra gli oggetti, su cui si costruisce il programma [JAVA]

**FUNZIONALI:** sono basati sul concetto matematico di funzione. [LISP]

**LOGICI:** l'istruzione è una clausola che descrive una relazione fra i dati. Non c'è un ordine prestabilito di esecuzione delle varie clausole, l'interprete trova l'ordine giusto [Prolog]

Linguaggi di von Neumann (esecuzione sequenziale di istruzioni che modificano lo stato della computazione)

26

## INTERPRETI E COMPILATORI

**L'interprete** è un programma che traduce, istruzione per istruzione, il programma sorgente. Ogni istruzione è sottomessa alla CPU appena è stata tradotta.

Anche se una stessa istruzione è contenuta 10 volte in un programma verrà tradotta ogni volta che si presenterà al traduttore.

**Il compilatore**, invece, traduce tutto il programma in una sola volta e poi lo sottomette alla CPU.

Se una stessa istruzione compare 10 volte, viene tradotta solo la prima volta e poi memorizzata in maniera tale che possa essere ripresa dal compilatore e inserita nelle restanti nove righe di programma



27

## PASSI: DAL PROBLEMA ALLA SOLUZIONE

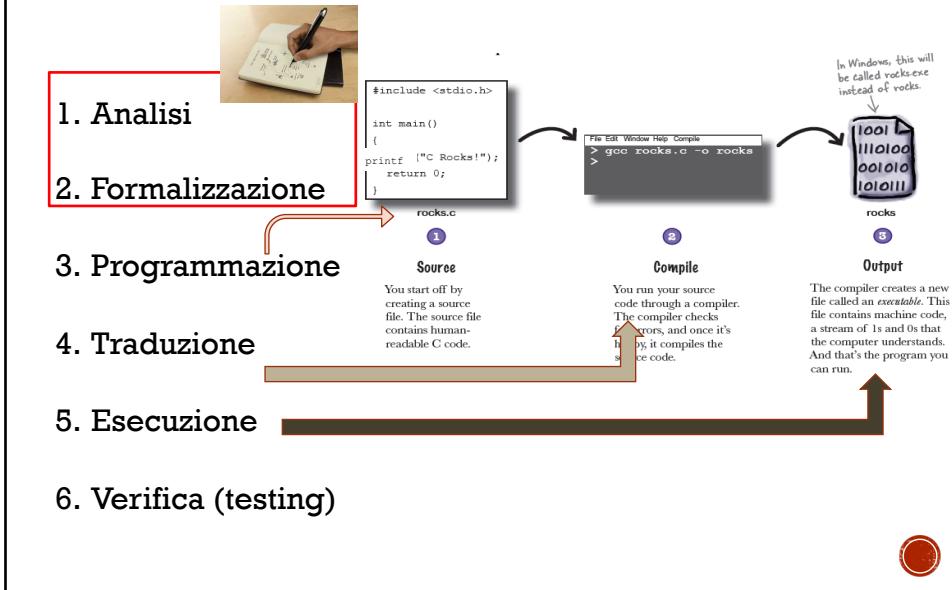


1. Analisi
2. Formalizzazione
3. Programmazione
4. Traduzione
5. Esecuzione
6. Verifica (testing)

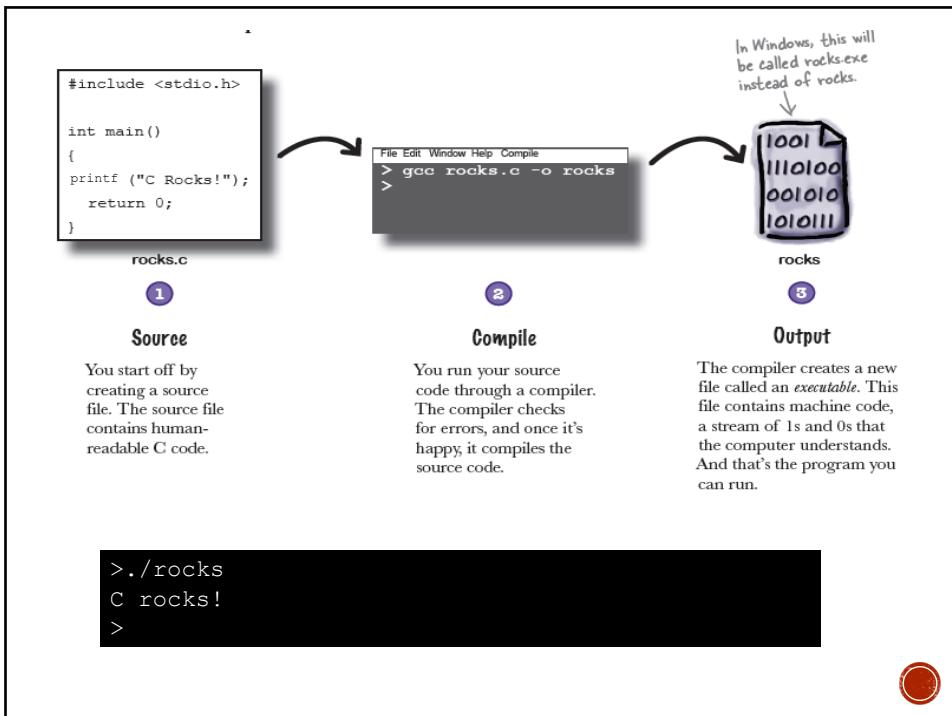


28

## PASSI: DAL PROBLEMA ALLA SOLUZIONE



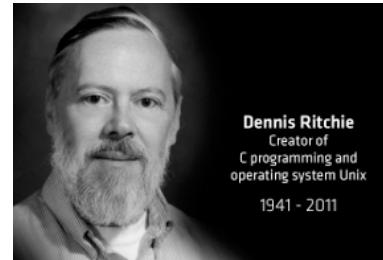
29



30

# BREVE STORIA DEL C

- C è un “figlio” di UNIX, sviluppato nei Bell Laboratories da Ken Thompson, Dennis Ritchie, e altri. UNIX è scritto in linguaggio assembly (!!!).
- Negli anni ‘60, Thompson sviluppò B, un piccolo linguaggio per SO, che, qualche anno dopo modificò, arrivando a NB.
- Il C fu poi sviluppato nel 1973. Vari standard (ANSI-C, C89, C99).



31

# BREVE STORIA DEL C

- C è un “figlio” di UNIX, sviluppato nei Bell Laboratories da Ken Thompson, Dennis Ritchie, e altri. UNIX è scritto in linguaggio assembly (!!!).
- Negli anni ‘60, Thompson sviluppò B, un piccolo linguaggio per SO, che, qualche anno dopo modificò, arrivando a NB.
- Il C fu poi sviluppato nel 1973. Vari standard (ANSI-C, C89, C99).

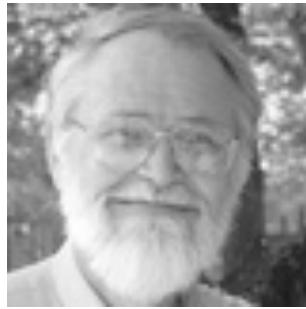
## Caratteristiche

Proprietà: a basso livello; “piccolo”; permissivo...

Punti di forza: efficiente; portatile; potente; flessibile; librerie standard; integrato con UNIX

Punti di debolezza: error-prone; criptico; poco modificabile

32



Brian Wilson Kernighan

**"[IL LINGUAGGIO DI  
PROGRAMMAZIONE C] È UNO  
STRUMENTO TAGLIENTE, CON CUI SI  
PUÒ CREARE UN PROGRAMMA  
ELEGANTE ED EFFICIENTE, O FARE  
UN BAGNO DI SANGUE."**



33

## OBFUSCATED C...

Il problema delle 8 regine:  
posizionare 8 regine sulla  
scacchiera in modo che  
nessuna possa attaccare l'altra

```
v,i,j,k,l,s,a[99];
main()
{
    for (scanf ("%d",&s);*a-s;v=a[j*=v]-a[i],k=i<s,j+=(v=j<s&&
(!k&&!printf(2+"\n\n%c"-(!l<<!j)," #Q"[l^v?(l^j)&1:2])&&
++l||a[i]<s&&v&&v-i+j&&v+i-j))&&(l=s),v||(i==j?a[i=k]=0:
++a[i])>=s*k&&++a[--i])
    ;
}
```

International Obfuscated C contest ([www.loccc.org](http://www.loccc.org))



34

## CLASSICI ERRORI

- “Paura” di scrivere il codice
- Scrivere TUTTO il programma e poi provarlo
- Dire: “Perché fa questo?” Risposta: perché glielo hai detto tu!!!!!!!!!
- Dire che “funziona” solo perché funziona su un caso



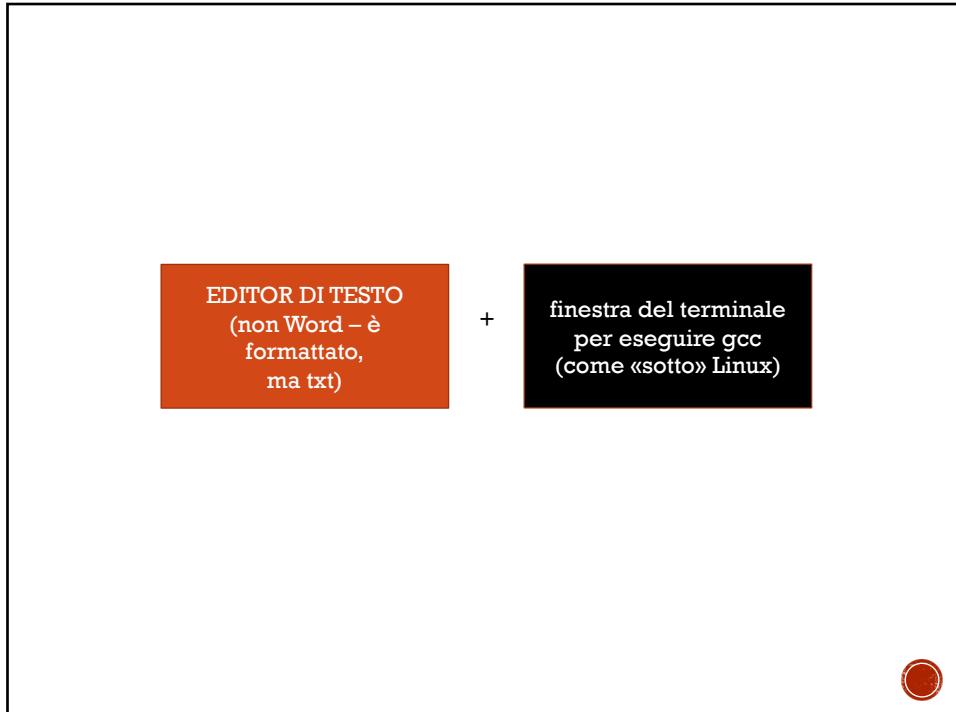
35

## NOI LAVOREREMO “TESTUALE”

- Dimenticatevi APP, ICONE, COLORI ☺
- Lavoriamo usando una “shell” (finestra), o meglio due shell
- E cosa faremo?



36



37

## IN LABORATORIO

- Vi ho mostrato come collegarvi alle macchine del laboratorio, quindi come:
  - aprire il terminale su Windows (WindowsPowerShell che ha già gcc sulle macchine del laboratorio)
  - aprire NotePad++ (disabilitando il completamento automatico)
  - fa corrispondere la directory dove si posiziona WindosPowerShell (o il terminale in generale) e la directory dove dovrete salvare il file .c scritto con l'editor NotePad++ (o gedit o TextEdit qualsiasi)
  - Salvare il file come .c

Tutto questo è scritto anche nella dispensa già caricata sul sito del corso.

NON RIMANDATE....

In the bottom right corner of the slide frame, there is a small red circular icon.

38

## IN LABORATORIO + A CASA...

Abbiamo solo rapidamente fatto insieme la fase di compilazione e di esecuzione.

Provate a fare voi da soli, come indicato in queste slide successive. Poi rifaremo rapidamente insieme in laboratorio, il prossimo venerdì.

A casa: esercitatevi con cygWin, installandolo come suggerito nella dispensa.

NON RIMANDATE....



39

## Esercizio 0

**Scrivere** un programma che stampi a video quanto segue  
(andando a capo alla fine). **Compilarlo. Eseguirlo.**

To C, or not to C: that is the question.

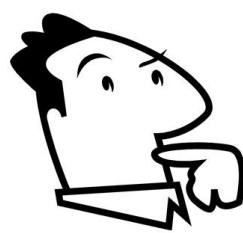


40

## Esercizio 0

**Scrivere** un programma che stampi a video quanto segue  
(andando a capo alla fine). **Compilarlo. Eseguirlo.**

To C, or not to C: that is the question.



Apro un Editor di testo e scrivo il codice

“Ogni tanto” salvo  
(ricorda: estensione .c e ricordati dove lo hai salvato!)

Apro il TERMINAL,  
eseguo con gcc

41

## Esercizio 0

**Scrivete** quanto segue nel file

```
#include <stdio.h>

int main(void)
{
    printf("To C, or not to C: that is the question.\n");
    return 0;
}
```

42

## Esercizio 0

Dopo averlo salvato (ad esempio, nominandolo esempio1.c), compilatelo dalla finestra del terminale. Non dovrebbero esserci errori. Sarà generato il file eseguibile, a.out [oppure a.exe sotto Windows]

```
>gcc esempio1.c
>
```

```
#include <stdio.h>

int main(void)
{
    printf("To C, or not to C: that is the question.\n");
    return 0;
}
```

43

## Esercizio 0

Potete eseguirlo, digitando

```
>./a.out
To C, or not to C: that is the question.
>
```

```
#include <stdio.h>

int main(void)
{
    printf("To C, or not to C: that is the question.\n");
    return 0;
}
```

44

## ESERCIZI DEL FINE SETTIMANA

- Troverete sul sito del corso (e sul mio sito personale unisa) un elenco di esercizi suggeriti.
- Fateli
- Se incontrate difficoltà, contattatemi.
- Non vi dimenticate che difficoltà tecniche per l'installazione di gcc sono affrontate nella guida caricata sul sito del corso (e sul sito personale unisa).



45

## IN QUESTA SETTIMANA

Approfittate per  
1) procurarvi libro,  
2) installare compilatore C

Settimana dal		al		Mese di		
Lunedì	Martedì	Mercoledì	Giovedì	Venerdì	Sabato	Domenica
-----	-----	-----	-----	-----	-----	-----
-----	-----	-----	-----	-----	-----	-----
-----	-----	-----	-----	-----	-----	-----
-----	-----	-----	-----	-----	-----	-----
-----	-----	-----	-----	-----	-----	-----
-----	-----	-----	-----	-----	-----	-----

[www.homemademamma.com](http://www.homemademamma.com)



46