Università degli Studi di Verona

Dipartimento di Biotecnologie

Laurea in Biotecnologie

Corso di Informatica 2014/2015

Esercizi (seconda parte)

Gennaio 2015 - Sergio Marin Vargas

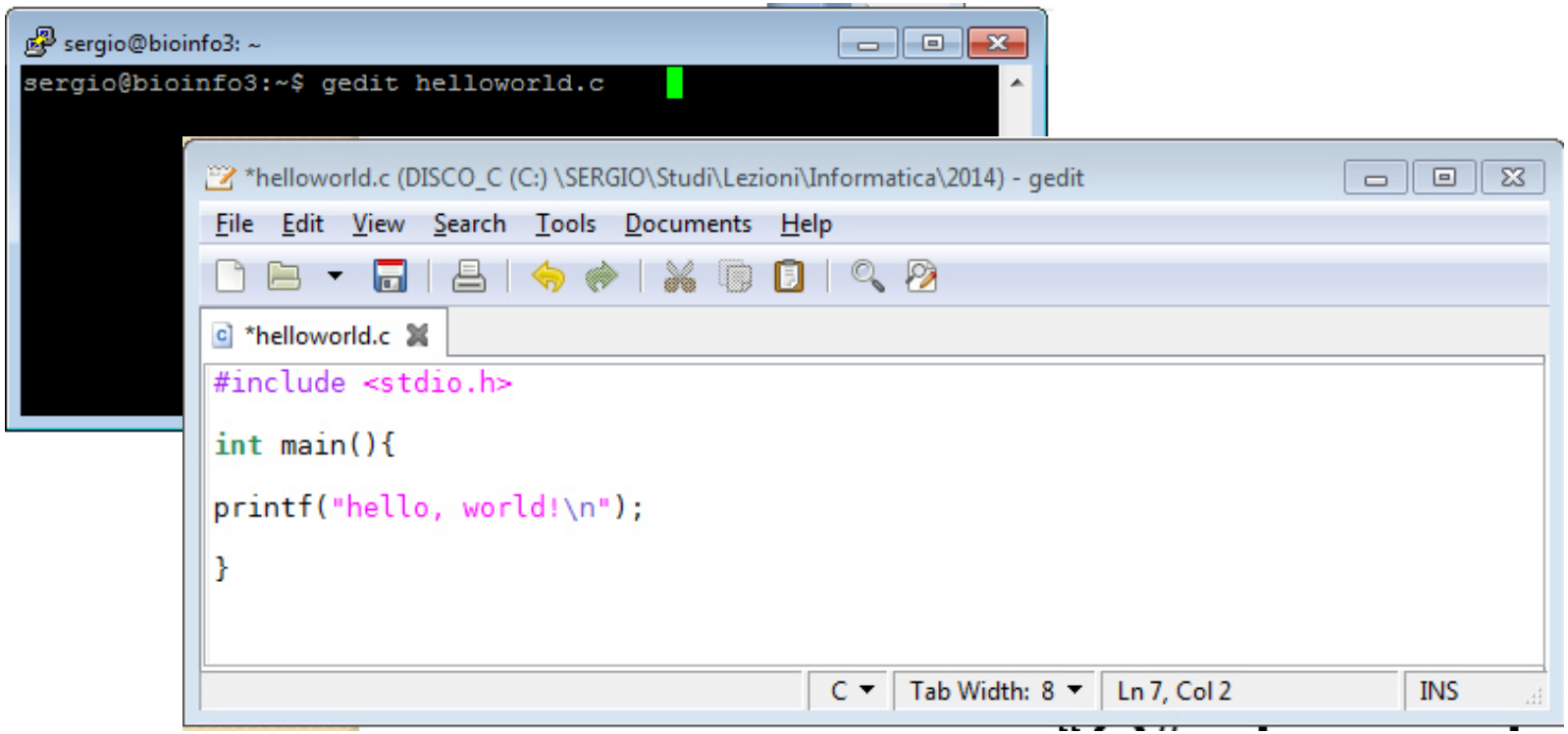
Scrivere un programma

Una volta che avete fatto il login

Lanciare il "terminale", all'interno del terminale scrivere:

gedit nome-programma.c

Il programma si scrive come in un qualsiasi editor di windows (tipo notepad).



Compilare un programma

- Aprite il terminale e posizionatevi nella directory dove si trova il programma (può essere ad esempio "Scrivania")

```
cd Scrivania
```

- Compilare il programma "**nome-programma.c**", dando all'eseguibile il nome "**nome-programma**"

```
gcc -o nome-programma nome-programma.c
```

- Il compilatore vi creerà un eseguibile "`nome-programma`", con già le autorizzazioni di esecuzione.

- Se ci sono errori, editare nuovamente il programma (con gedit), modificarlo, correggere gli errori e compilarlo nuovamente.

- Per vedere la lista di files in una directory, utilizzare il commando

```
ls -l
```

Eseguire un programma

- Per eseguire il programma con nome "**nome-programma**" creato a partire dal sorgente "**nome-programma.c**":

```
./nome-programma
```

- Poi fare Invio.

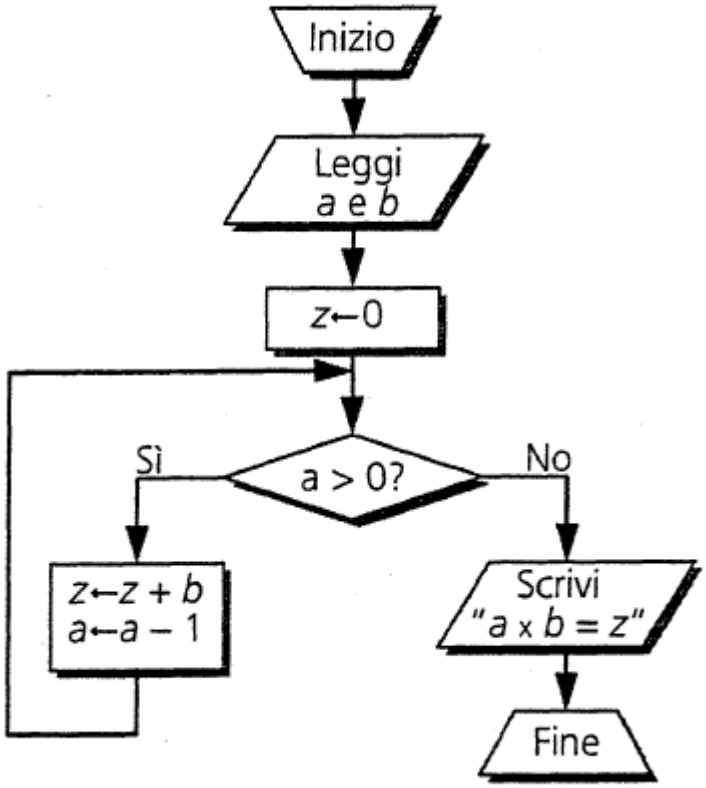
Problema 4 (a)

Dati in ingresso due numeri A e B, si realizzi un programma "multAxB" che calcoli il prodotto di "A" e "B" senza utilizzare l'operazione di moltiplicazione.

Algoritmo:

P1. Leggere un primo valore in input e assegnarlo alla variabile A
P2. Leggere un secondo valore in input e assegnarlo alla variabile B
P3. Inizializzare una variabile z = 0 (conterrà il prodotto di A x B)
P4. Mentre A > 0 eseguire i passi da P5 a P6
P5. Sottrarre 1 da A
P6. Sommare B alla variabile Z
P7. Visualizzare "Il prodotto dei due numeri è " Z
P8. Fine del programma

Problema 4 (a) - Algoritmo



$$z_0 = 0$$
$$z_1 = z_0 + b = b$$
$$z_2 = z_1 + b = 2 * b$$
$$\ldots$$
$$z_a = z_a - 1 + b = a * b$$

Problema 4 (a) - Programma

```
#include <stdio.h>
int main() {
int a, b;
int z;
    printf("\nInserisci il valore di A\n");
    scanf("%d", &a);

    printf("1nInserisci il valore di B\n");
    scanf("%d", &b);

    z = 0;
    while (a > 0) {
        a = a - 1;
        z = z + b;
    }

    printf("Il prodotto di A x B e' %d\n", z);
}
```

Problema 4 (b)

Dati in ingresso due numeri A e B, si realizzi un programma "multAxB2" che calcoli il prodotto di "A" e "B" utilizzando l'operazione di moltiplicazione.

Algoritmo:

P1. Leggere un primo valore in input e assegnarlo alla variabile A

P2. Leggere un secondo valore in input e assegnarlo alla variabile B

P3. Calcolare il prodotto nella variabile Z = A * B

P4. Visualizzare "Il prodotto di " A " x " B " è " Z

P5. Fine del programma

Problema 4 (b) - Programma

```
#include <stdio.h>
int main() {
int a, b;
int z;
    printf("\nInserisci il valore di A\n");
    scanf("%d", &a);

    printf("1nInserisci il valore di B\n");
    scanf("%d", &b);

    z = a * b;

    printf("Il prodotto di %d x %d e' %d\n", a, b, z);
}
```

Problema 5 (a)

Dati in ingresso un numero N, si realizzi un programma "sumNnum" che calcoli la somma dei numeri da 1 a N.

Algoritmo:

P1. Leggere il valore di "N" in input e assegnarlo alla variabile N

P2. Inizializzare variabile risultato ris = 0 e contattore i = 0

P3. Mentre che i non sia maggiore di N eseguire P4 e P5

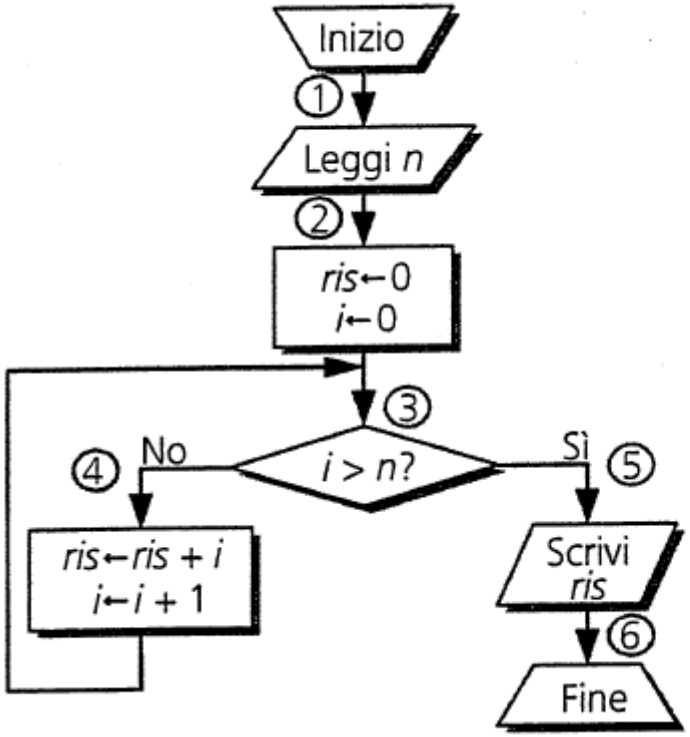
P4. ris = ris + i

P5. i = i + 1

P6. Visualizzare "La somma dei primi " N " numeri è " ris

P7. Fine del programma

Problema 5 (a) - Algoritmo



T	posiz.	*n*	*i*	*ris*	note
t_{01}	①	??	??	??	Variabili non ancora definite
t_{02}	②	4	??	??	Letto il valore 4 e inserito in *n*
t_{03}	③	4	0	0	$i < n \Rightarrow$ posiz. 4
t_{04}	④	4	0	0	
t_{05}	③	4	1	0	$i < n \Rightarrow$ posiz. 4
t_{06}	④	4	1	0	
t_{07}	③	4	2	1	$i < n \Rightarrow$ posiz. 4
t_{08}	④	4	2	1	
t_{09}	③	4	3	3	$i < n \Rightarrow$ posiz. 4
t_{10}	④	4	3	3	
t_{11}	③	4	4	6	$i < n \Rightarrow$ posiz. 4
t_{12}	④	4	4	6	
t_{13}	③	4	5	10	$i < n \Rightarrow$ posiz. 5
t_{14}	⑤	4	5	10	
t_{15}	⑥	4	5	10	Stampato risultato (10)

Problema 5 (a) - Programma

```
#include <stdio.h>
int main() {
int n;
int ris, i;
    printf("\nInserisci il valore di N\n");
    scanf("%d", &n);

    ris = 0;
    i = 0;
    while (! (i > n) ) {
        ris = ris + i;
        i = i + 1;
    }

    printf("La somma dei primi %d numeri e' %d\n", n, ris);
}
```

Problema 5 (b)

Dati in ingresso un numero N, si realizzi un programma "sumNnum2" che calcoli la somma dei numeri da 1 a N senza utilizzare nessun ciclo (while, until o for).

Algoritmo:

Supponiamo N = 10.

Sum = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10

Sum = 10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1

------ ----------

2Sum = 11 + 11 + 11 + 11 + 11 + 11 + 11 + 11 +11 + 11 (N volte "N+1")

2Sum = N(N+1)

→ **Sum = N(N+1) / 2**

Problema 5 (b) - Algoritmo

Algoritmo:

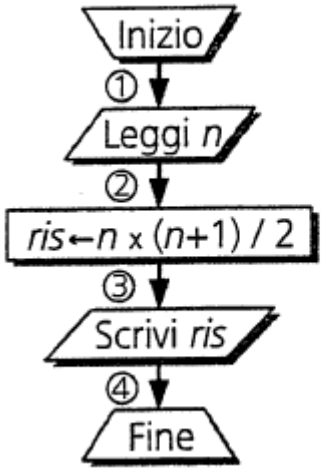
P1. Leggere il valore di "N" in input e assegnarlo alla variabile N

P2. Calcolare la soma dei primi N numeri con la formula

ris = N (N + 1) / 2

P3. Visualizzare "La somma dei primi " N " numeri è " ris

P4. Fine del programma



T	posiz.	*n*	*ris*	note
t_{01}	①	??	??	Variabili non ancora definite
t_{02}	②	4	??	Letto il valore 4 e inserito in *n*
t_{03}	③	4	10	Calcolato il risultato 10 = 4 x 5/2
t_{04}	④	4	10	Stampato risultato (10)

Problema 5 (b) - Programma

```
#include <stdio.h>
int main() {
int n;
int ris;
    printf("\nInserisci il valore di N\n");
    scanf("%d", &n);

    ris = n * (n + 1) / 2;

    printf("La somma dei primi %d numeri e' %d\n", n, ris);
}
```

Problema 6

Dato in ingresso un numero N (N > 2), si realizzi un programma con nome "fibonacci" che visualizzi i primi "N" numeri della serie di Fibonacci.

Serie di Fibonacci

$$F_n = 1 \qquad n = 1,2$$
$$F_n = F_{n-1} + F_{n-2} \qquad n > 2$$

Sequenza:	1	2	3	4	5	6	7	8	9	10	…
Serie:		1	1	2	3	5	8	13	21	34	55 …

Problema 6 - Programma

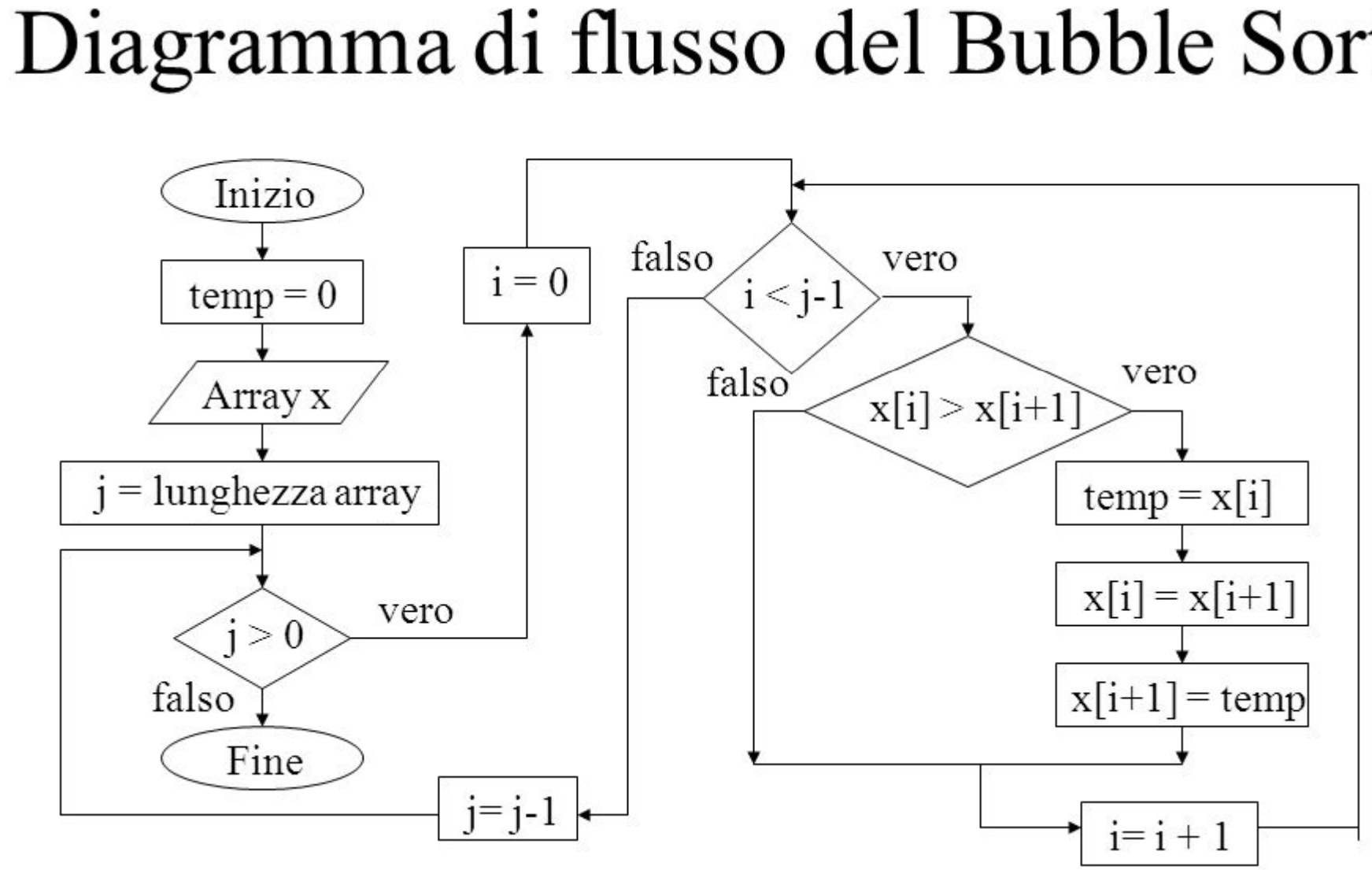
```
#include <stdio.h>
int main() {
int n;
int i, fn, serie_2, serie_1, serie;
    printf("\nInserisci il valore di N\n");
    scanf("%d", &n);

    printf("\nElementi della serie di Fibonacci\n");

    i = 0;

    i++;
    fn = 1;
    serie_2 = fn;
    printf("Elemento %d = %d\n", i, serie_2);

    i++;
    fn = 1;
    serie_1 = fn;
    if (n > 1) {
      printf("Elemento %d = %d\n", i, serie_1);
    }
```

```
    while (i < n) {
      i++;
      serie = serie_1 + serie_2;
      printf("Elemento %d = %d\n", i, serie);
      serie_2 = serie_1;
      serie_1 = serie;
    }
}
```

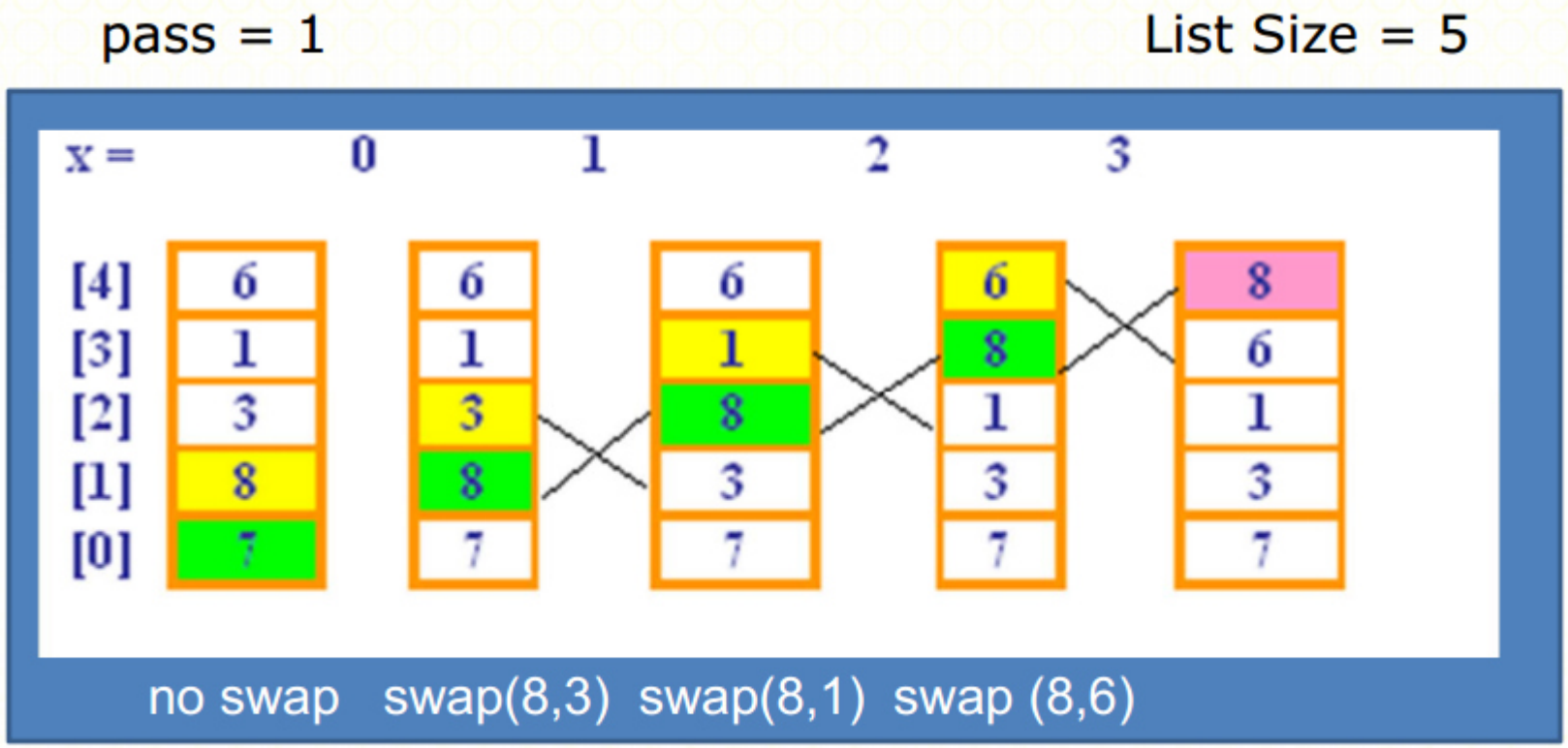
Problema 7

Dati in ingresso N numeri (N < 100), realizzare un programma con nome "sortArray" che verifichi che la dimensione non superi quella richiesta, quindi che legga gli N numeri, li carichi in un Array, li ordini in maniera crescente e li visualizzi già ordinati.

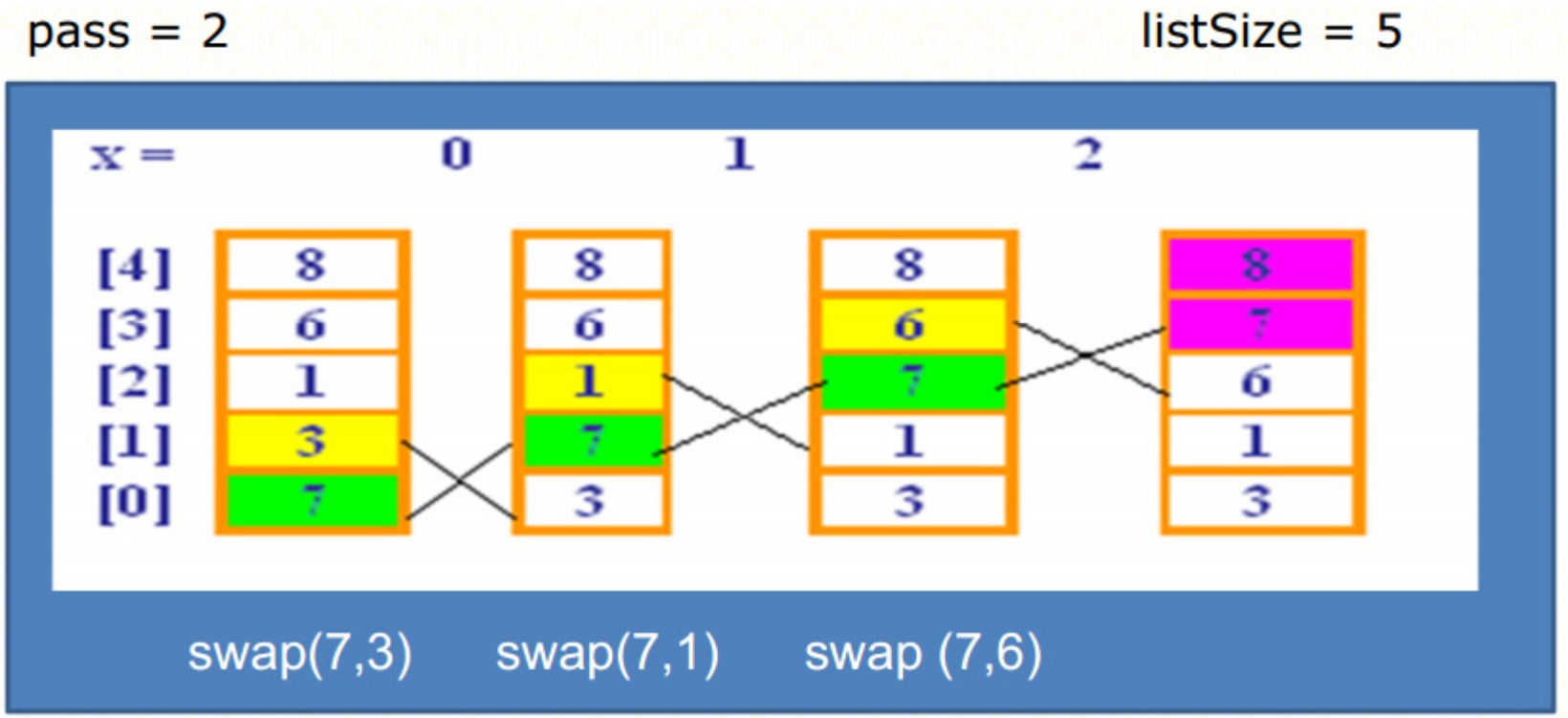
Algoritmo Bubble SORT



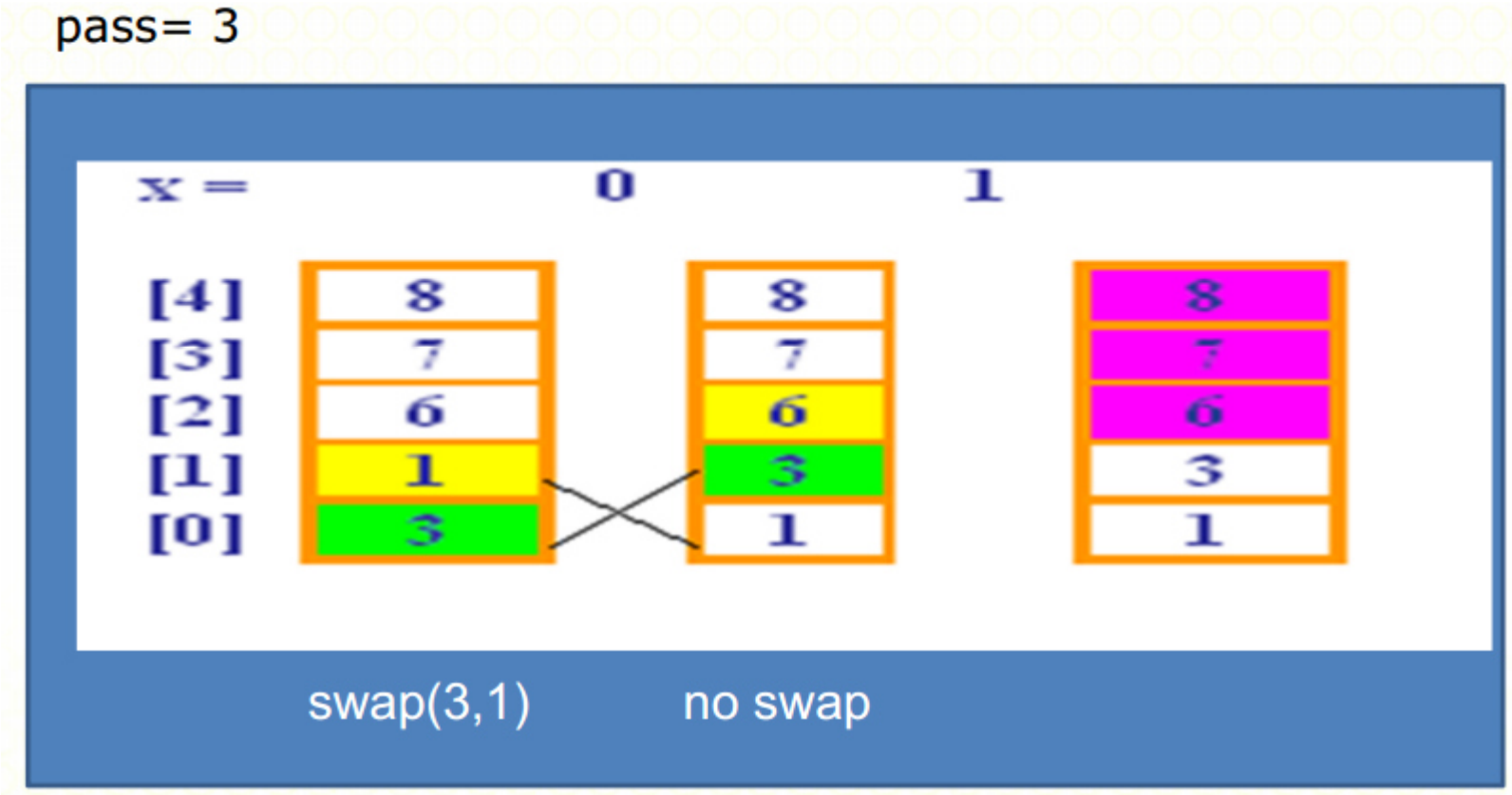
Analisi del Bubble Sort (Passo 1)



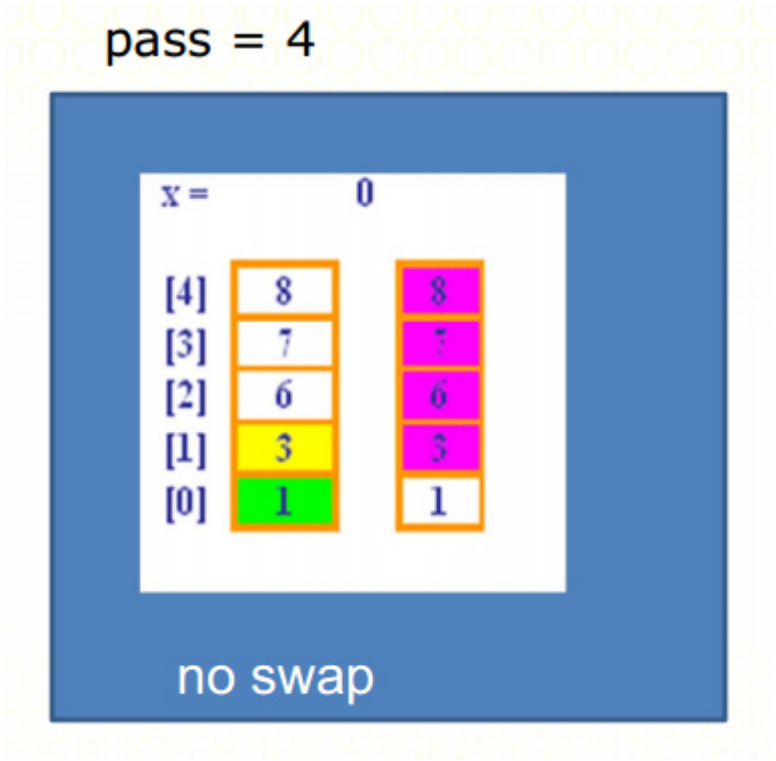
Analisi del Bubble Sort (Passo 2)



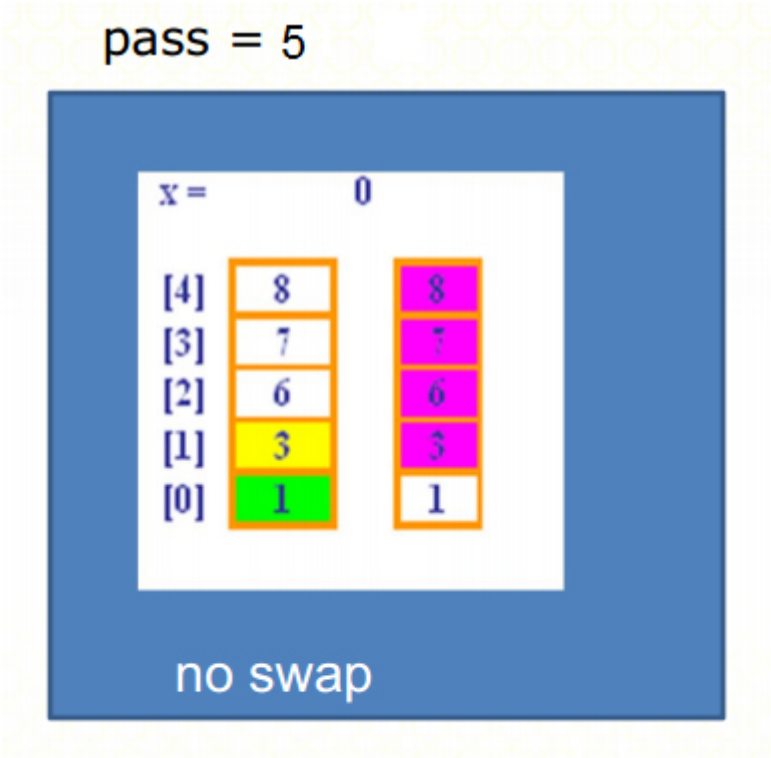
Analisi del Bubble Sort (Passo 3)



Analisi del Bubble Sort (Passo 4)



Analisi del Bubble Sort (Passo 5)



Nell'ultimo passo l'array sarà ordinato !!!

Algoritmo Problema 7

P1. Leggere "quanti numeri" e assegnarlo alla variabile N

P2. Inizializzare contatore i = 0

P3. Mentre i < N fare i passi da P4 a P5:

P4. Leggere il valore i-esimo e assegnarlo all'elemento i dell'array X

P5. i = i + 1

P6. Ordinare l'array X in maniera crescente (algoritmo di SORT)

P7. Inizializzare contatore i = 0

P8. Ciclo variabile i da 0 a N, iterando i passi da P9 a P10:

P9. Visualizza "Elemento(" + i+1 + ") = " X[i]

P10. i = i + 1

P11. Fine del programma