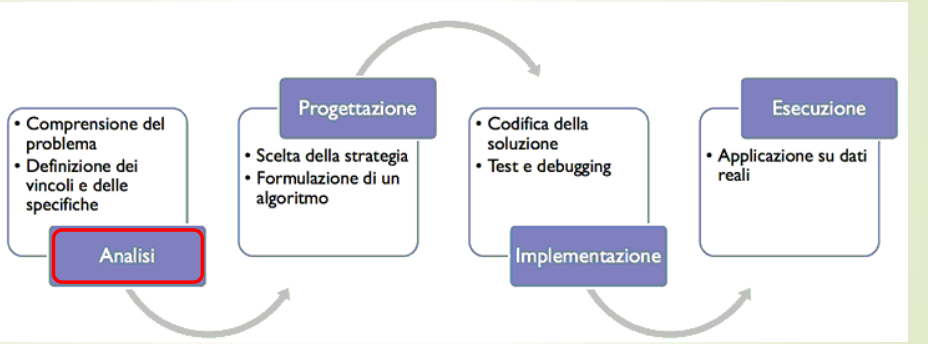
Sviluppo di Programmi

Analisi e Progettazione
Selection Sort

Fasi dello sviluppo

Fasi dello sviluppo



Analisi

- Specifica di *cosa fa* il programma. Individuazione di:
  - **Dati di ingresso e loro vincoli**
  - **Dati di uscita e loro vincoli**

Analisi
Vincoli

- ***Precondizione***: condizione definita sui dati di ingresso che deve essere soddisfatta affinché la funzione sia applicabile
- ***Postcondizione***: condizione definita su dati di uscita e dati di ingresso e che deve essere soddisfatta al termine dell'esecuzione del programma
  - **definisce cosa sono i dati di output in funzione di quelli di input …**

Analisi
Dizionario dei Dati

- Buona norma utilizzare un ***dizionario dei dati*** da arricchire durante le varie fasi del ciclo di vita
  - **Una tabella il cui schema è:**
    - ***Identificatore, Tipo, Descrizione***
    - **La descrizione serve a specificare meglio l'identificatore e a descrivere il contesto in cui il dato viene usato**

Analisi

Un esempio: ordinamento di una sequenza di interi

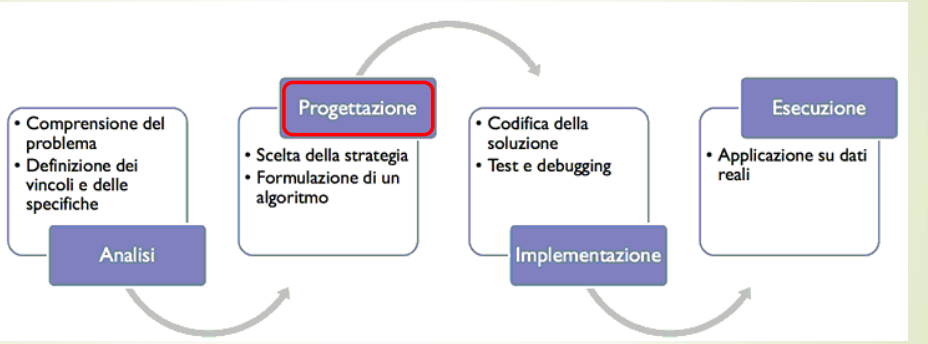
- Dati di ingresso: sequenza s di n interi
  - Precondizione: $n > 0$
- Dati di uscita: sequenza s1 di n interi
  - Postcondizione: s1 è una permutazione di s dove

$$\forall i \in [0,\ n-2],\ s1_i \leq s1_{i+1}$$

Dizionario dei dati

Identificatore	Tipo	Descrizione
s	sequenza	sequenza di interi in input
s1	sequenza	sequenza di interi di output
n	intero	numero di elementi nella sequenza
i	intero	indice per individuare gli elementi nella sequenza

Fasi dello sviluppo



Progettazione

- Definizione di *come* il programma effettua la trasformazione specificata
- Progettazione dell'algoritmo per raffinamenti successivi (stepwise refinement)
  - Definizione degli step
  - Decomposizione funzionale

Progettazione

Esempio: ordinamento di una sequenza di interi

1. Input sequenza s in un array a di dimensione n
2. Ordina array a di dimensione n
   NB: per motivi di efficienza decidiamo di usare un unico array di input e output
3. Output sequenza s1 contenuta in array a di dimensione n

Raffinamento del programma principale: definiamo delle funzioni corrispondenti agli step individuati

- **input_array(a, n)**
- **ordina_array(a, n)**
- **output_array(a, n)**

Per ognuna:
- **specifica**
- **progettazione**
- **codifica e verifica**

11

Funzione ordina_array

1. **Analisi**: Specifica simile a quella del programma principale, ma introduciamo l'array …
2. **Progettazione**: scegliamo come strategia di ordinamento *Selection Sort* …
3. **Codifica e verifica** …

12

Analisi

Funzione ordina_array

- Dati di ingresso: **array a di interi di dimensione n**
  - Precondizione: **n > 0**
- Dati di uscita: **array a di interi di dimensione n**
  - Postcondizione: l'array a in output contiene una permutazione degli elementi dell'array a in input t.c. ∀ i ∈ [0, n-2], a[i] <= a[i+1]

Dizionario dei dati

Identificatore	Tipo	Descrizione
a	array	array di interi
n	intero	dimensione dell'array
i	intero	indice per individuare gli elementi dell'array

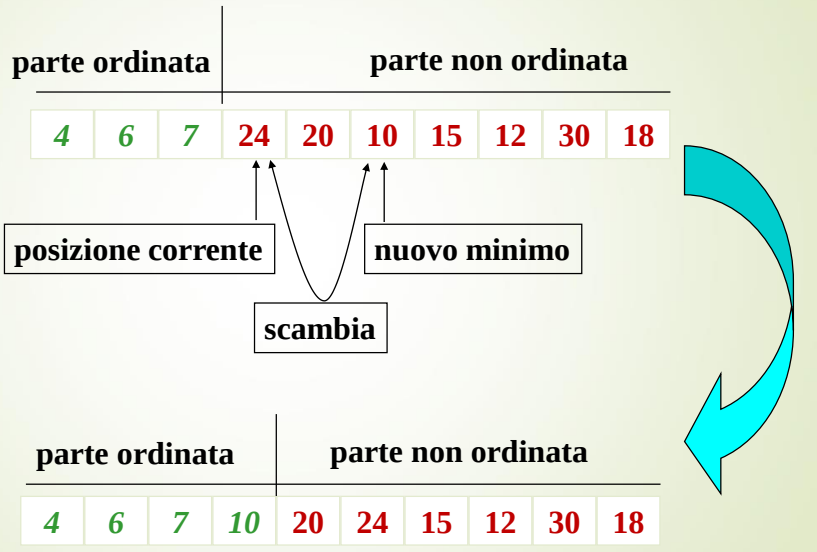
13

Selection Sort

- Effettua una visita totale delle posizioni dell'array
  - ***visita totale*:** visitati in sequenza tutti gli elementi dell'array
- Per ogni posizione visitata individua l'elemento che dovrebbe occupare quella posizione
  - In questo modo, se *i* è la posizione corrente (0 <= *i* < *n*), tutti gli elementi nelle posizioni comprese tra 0 ed *i*-1 rispettano l'ordinamento;
  - L'elemento che deve occupare la posizione *i* sarà il minimo tra quelli nelle posizioni comprese tra *i* ed *n*-1;

14

Selection Sort



15

Progettazione
Funzione ordina_array

- for(i = 0; i < n-1; i++)
  1. Individua la posizione p dell'elemento minimo compreso tra le posizioni i e n-1 dell'array a
  2. Scambia gli elementi di a di posizioni i e p

Raffinamento del programma principale: definiamo delle funzioni corrispondenti agli step individuati

- minimo(a, n)
- scambia(i, j)

16

Analisi
Funzione minimo

- Dati di ingresso: array a di interi di dimensione n
  - Precondizione: n > 0
- Dati di uscita: min
  - Postcondizione: ∀ i ∈ [0, n-1], a[min] <= a[i]

Dizionario dei dati

Identificatore	*Tipo*	*Descrizione*
a	array	array di interi
n	intero	dimensione dell'array
min	intero	indice dell'elemento minimo
i	intero	indice per individuare gli elementi dell'array

17

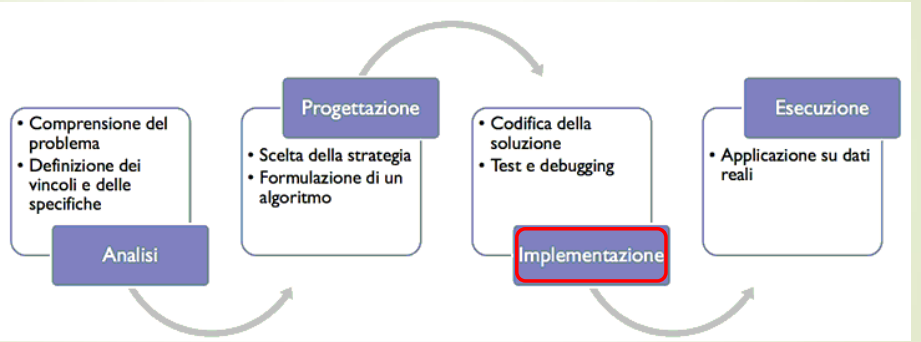
Progettazione

Funzione minimo

```
min = 0;
for(i = 1; i < n; i++)
    if(a[i] < a[min])
        min = i;
return(min);
```

18

Fasi dello sviluppo



Implementazione

- Codifica dell'algoritmo nel linguaggio scelto
- Verifica (testing) del programma (individuazione dei malfunzionamenti)
  - **Scelta dei casi di prova**
  - **Esecuzione del programma**
  - **Verifica dei risultati rispetto ai *risultati attesi***
- *Utilizzo del software di base e di un ambiente di sviluppo …*

Codifica


```
void selection_sort(int *arr, int n){
   int i;
   for(i=0; i<n-1; i++){
         int min = minimo(&arr[i], n-i) + i;
         scambia(&arr[i], &arr[min]);
   }
}
```

```
int minimo(int *arr, int n){
   int min = 0, i;
   for(i=1; i<n; i++)
         if (arr[min] > arr[i])
               min = i;
   return min;
}
```

```
void  scambia(int * x, int * y)
{
   int temp = *x;
   *x = *y;
   *y = temp;
}
```