Universita' degli Studi di Salerno
Dipartimento di Informatica, Scienza dell'informazione
Corso di Programmazione e Strutture Dati (PsD)

Lezione No. 1

Salvatore Pisu
(Matricola: NF12100068)
salvatore.pisu1@studenti.unisa.it

Anno Accademico 2025/2026

Sommario

Il presente documento analizza le fasi metodologiche dello sviluppo software, distinguendo tra l'astrazione dell'algoritmo e l'implementazione del programma. Viene approfondito il ciclo di vita del software, dal contratto logico tra precondizioni e postcondizioni fino alla progettazione di algoritmi di ricerca e ordinamento mediante raffinamenti successivi.

Indice

1 Fondamenti dello Sviluppo Software

Sviluppare programmi non consiste esclusivamente nella scrittura di codice. È necessario progettare soluzioni che siano simultaneamente corrette ed efficienti. Il corso si focalizza sulla risoluzione di problemi, sulla rappresentazione dei dati e sulla progettazione di algoritmi.

Algoritmo ($\mathcal{A}$) vs Programma ($\mathcal{P}$)

- **Algoritmo ($\mathcal{A}$):** Descrizione astratta di una soluzione indipendente dal linguaggio di programmazione. È composto da passi logici finiti e non ambigui e descrive *cosa* fare.
- **Programma ($\mathcal{P}$):** Implementazione concreta di un algoritmo in un linguaggio di programmazione. È una sequenza di istruzioni eseguibili che opera la trasformazione fondamentale:

$$\text{Input} \xrightarrow{\mathcal{P}} \text{Output}$$

Deve risultare corretto, comprensibile e verificabile.

2 Ciclo di Vita e Analisi e Specifica

Lo sviluppo segue una progressione logica: Analisi → Progettazione → Codifica → Verifica.

2.1 Contratto Logico

L'Analisi definisce **cosa** fa il programma tramite i dati di ingresso/uscita e le condizioni logiche.

Precondizioni e Postcondizioni

La correttezza è garantita dall'implicazione: *Se la precondizione è vera e il programma viene eseguito, allora deve valere la postcondizione.*

- **Precondizione (π):** Condizione sui dati di ingresso necessaria per l'esecuzione.
- **Postcondizione (ϕ):** Stato finale dei dati di uscita in funzione dell'input.

2.2 Dizionario dei Dati

È una tabella formale che mappa identificatore, tipo e descrizione del contesto.

Identificatore	Tipo	Descrizione
`s, s1`	sequenza	Interi in input e output (permutazione ordinata).
`n`	intero	Numero di elementi nella sequenza ($n > 0$).
`i`	intero	Indice per la scansione.

3 Algoritmi Elementari

3.1 Selection Sort

Effettua una visita totale dell'array. Per ogni posizione i, individua il minimo tra le posizioni `i` e `n-1` e lo scambia con l'elemento in posizione `i`.

3.2 Ricerca Binaria

Applicabile solo su array ordinati. Divide lo spazio di ricerca a metà. Nel caso peggiore si visitano solo $\log_2 n$ elementi.

Confronto Strategie di Ordinamento

Algoritmo	Meccanismo	Caratteristiche
Selection	Selezione iterativa del minimo.	In-place ($\mathcal{O}(1)$), pochi scambi.
Insertion	Inserimento in posizione corretta.	Efficiente per array quasi ordinati.
Bubble	Scambio di coppie adiacenti.	Complesso $\mathcal{O}(n^2)$, facile da implementare.

4 Fondamenti Tecnici: Puntatori

Un puntatore rappresenta l'indirizzo di memoria di un altro dato. Deve essere inizializzato prima dell'uso.

Funzione `scambia`

Utilizza il passaggio per riferimento per scambiare i valori di due variabili.

```
1        void scambia(int *x, int *y) {
2            int temp = *x; // Salvataggio valore
3            *x = *y;       // Scambio valori
4            *y = temp;     // Ripristino da temp
5        }
```