Programmazione e Strutture Dati (PR&SD)

I° ANNO – Informatica

classe 2, matricole congrue a 1 modulo 3

Lezione 0: Introduzione al Corso

Docenti:

Prof. Maurizio TUCCI

Dott. Fabio Narducci

Presentazione del Corso

- **Orario Lezioni**
  - Lunedì 9:00 - 11:00
  - Mercoledì 9:00 - 11:00
  - Venerdì 9:00 - 13:00

- **Periodo Didattico**

  1 Marzo 2021 – 4 Giugno 2021

- **Orario Ricevimento**
  - **Prof. TUCCI**
    - Lunedì 11:00-13:00 + contattare il docente: mtucci@unisa.it
    - Martedì 12:00-13:00

Contenuti del Corso

- Obiettivi formativi
  - APPROFONDIRE GLI ASPETTI DELLA PROGRAMMAZIONE PROCEDURALE RELATIVI ALLA ASTRAZIONE DATI.
  - INTRODURRE STRUTTURE DATI FONDAMENTALI, COME STACK, CODE, LISTE, ALBERI E TABELLE HASH
  - APPROFONDIRE ALCUNE TECNICHE DI PROGETTAZIONE E REALIZZAZIONE DI PROGRAMMI, UTILIZZANDO SOLUZIONI ITERATIVE E RICORSIVE.
  - ANALIZZARE PROBLEMI TIPICI E REALIZZARE APPLICAZIONI CHE LI RISOLVANO PROGETTANDO E REALIZZANDO ALGORITMI E STRUTTURE DATI IN LINGUAGGIO C. REALIZZAZIONE DI PROGETTI SOFTWARE IN C DI PICCOLE DIMENSIONI.
  - SELEZIONARE GLI ALGORITMI E LE STRUTTURE DATI ADEGUATE A SUPPORTARE UN'APPLICAZIONE, SULLA BASE DELLE SPECIFICHE ESIGENZE APPLICATIVE INDIVIDUANDO APPROPRIATE SOLUZIONI ITERATIVE O RICORSIVE PER GESTIRE UNO SPECIFICO PROBLEMA DI PROGRAMMAZIONE.

Contenuti del Corso

- Argomenti trattati
  - COMPLEMENTI DI PROGRAMMAZIONE IN C: STRUTTURE A PUNTATORI, RICORSIONE
  - TIPI DI DATI ASTRATTI (ADT): PROGETTO E REALIZZAZIONE. INTERFACCIA E IMPLEMENTAZIONE DI UN TIPO DI DATI.
  - SPECIFICA SINTATTICA E SEMANTICA DEGLI ADT DI BASE: LISTE, STACK, CODE, ALBERI BINARI, TABELLE HASH
  - IMPLEMENTAZIONI IN LINGUAGGIO C.

Presentazione del Corso

- Lezioni frontali (a distanza)
  - Approfondimento di principi e tecniche della programmazione procedurale
  - Aspetti formali e teorici sui tipi di dati astratti
  - Progettazione e realizzazione in C di strutture dati

- Lezioni pratiche di laboratorio (sempre a distanza)
  - Progettazione, Compilazioni, Testing di programmi C scritti per realizzare ed utilizzare le strutture dati

Presentazione del Corso

Libri di Testo consigliati

ROBERT SEDGEWICK, “ALGORITMI IN C” 4/ED”, PEARSON,
ISBN: 9788891900746

Appunti e dispense del Corso (a cura del docente)

Piattaforma e-learning del Dipartimento di Informatica

Canale Teams del corso

Modalità di Esami in presenza (se consentiti)

- 2 Prove in itinere a carattere concettuale-pratico tenute in laboratorio durante il corso
  - 1a prova: metà Aprile
  - 2a prova: fine Maggio
    (La media ponderata delle prove in itinere determina il voto di ammissione all'orale)

- 1 prova di laboratorio per ciascun appello di esame
  - Per gli studenti che non hanno superato le 2 prove in itinere

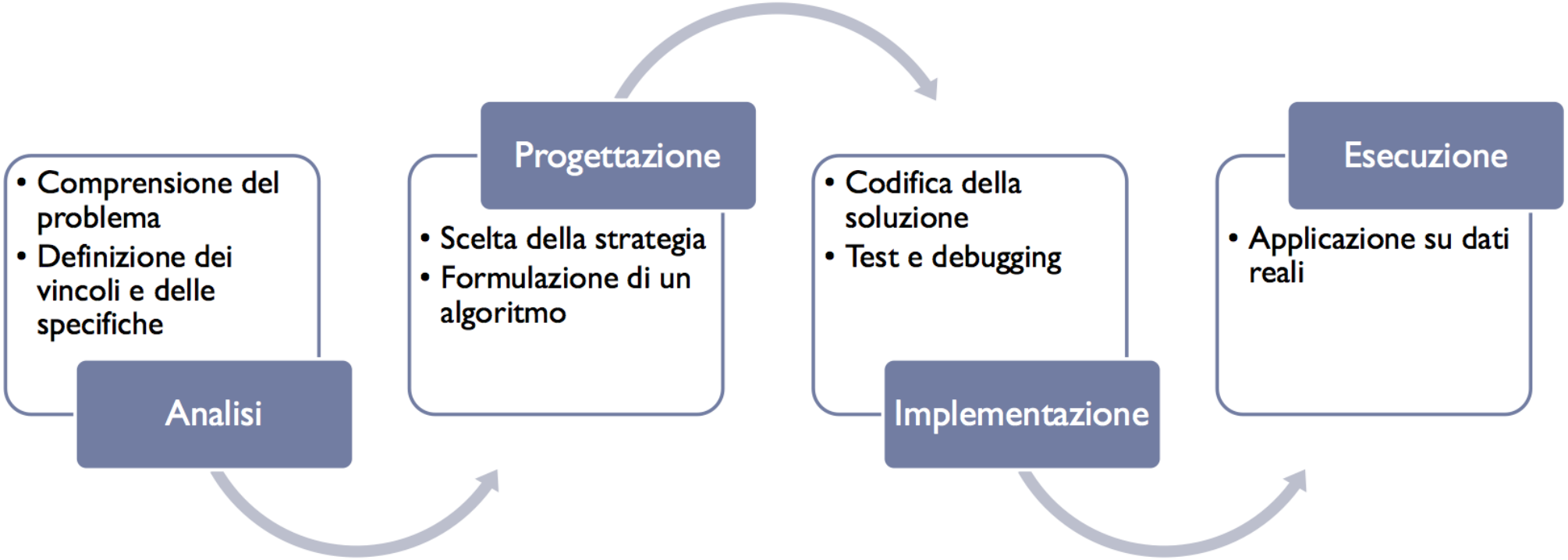
- Prova orale

Modalità di Esami a distanza (al momento previsti)

- 1 prova scritta per ciascun appello di esame, in forma di test a risposta multipla svolto a distanza

- Prova orale

Che cosa impariamo durante il corso?



Algoritmi e Programmi

Problema → *Algoritmo* → *Programma*

- Definizione del *problema*
  - Dati iniziali (input)
  - Dati intermedi (se necessari)
  - Dati finali (output)
  - Definizione di un insieme di passi logici che trasformano i dati iniziali in dati finali
- Il risultato è un *algoritmo*, un insieme finito di istruzioni che, se eseguite ordinatamente, sono in grado di risolvere il problema di partenza.

Dati

- I **dati** di un programma sono i valori assunti dalle sue variabili

- Un **tipo** di dati è definito da un **dominio di valori** e un insieme di **operazioni** previste su quei valori

  esempio: il tipo interi può essere definito da un intervallo di valori interi $[-2^{m-1}, 2^{m-1} - 1]$ e dalle operazioni aritmetiche/logiche elementari {+, -, *, /, %, ==, !=, >, <, <=, >=}

Dati

- Tipi di dati **primitivi**: forniti direttamente dal linguaggio di programmazione
  - int, char, float, double
- Dati **aggregati**:
  - array, strutture, enumerazioni, unioni, file
- Puntatori

Costruzione di nuovi Tipi di Dati

- **Specifica** e **implementazione**

- **Specifica**:
  - Definizione del dominio dei valori
  - Definizione dell'insieme degli operatori

- **Implementazione:**
  - Codifica di quanto definito nella specifica, usando primitive e costrutti di un linguaggio di programmazione

Strutture Dati

- Le strutture di dati sono collezioni di dati organizzati in maniera strutturata
- Sono caratterizzate da:
  - un modo sistematico di organizzare l'insieme dei dati
  - un insieme di operatori che permettono di manipolare la struttura
- Alcune tipologie di strutture di dati:
  - Lineari / Non lineari (presenza di una sequenza)
  - Statiche / Dinamiche (variazione di dimensione, contenuto)
  - Omogenee / Disomogenee (rispetto ai dati contenuti)

Strutture Dati lineari

- I dati sono organizzati in maniera lineare (cioè in sequenza monodimensionale)
- La loro dimensione è fissata oppure varia dinamicamente
- E’ possibile accedere/aggiungere/togliere elementi in determinate posizioni (inizio, fine, posizioni intermedie)
- Alcuni esempi: pile, code, liste

Strutture Dati non lineari

- I dati sono organizzati in maniera non sequenziale
  - Insiemi di dati non ordinati (dizionari, tabelle hash)
  - Dati organizzati in maniera gerarchica (alberi)
  - Dati organizzati in reti di nodi/archi (grafi)
- Spesso la loro dimensione varia dinamicamente
- E' possibile accedere/aggiungere/togliere elementi in determinate posizioni

Astrazione dati e programmazione modulare

- Realizzare programmi suddivisi in *moduli*
  - Un modulo è una porzione di programma (funzioni, dichiarazioni di tipi, variabili, macro, etc.) che svolge una parte dei compiti in maniera quanto più possibile indipendente dagli altri
- *Astrazione* ed *Information hiding*
  - *Interfaccia* separata dalla *codifica*
  - Per usare un modulo non è necessario conoscere i dettagli della codifica
- Indipendenza dei moduli
  - Organizzazione strutturata del programma
  - Compilazione separata
  - Manutenibilità locale

Ricorsione

- I dati sono organizzati in strutture definite in termini di se stesse (ricorsivamente)
  - Una lista è una sequenza di elementi che
    - o è vuota
    - oppure ha un primo elemento e i rimanenti elementi costituiscono a loro volta una lista
- Le operazioni sui dati sono definite in maniera ricorsiva
  - L'operatore fattoriale(n), denotato come n!, è definito per tutti gli interi n>=1 come:
    - n! è uguale a 1 se n è uguale a 1
    - n! è uguale a n * (n-1)! se n è maggiore di 1
- I tipi di dati e i loro operatori sono implementati con strutture/sottoprogrammi ricorsivi

```
int  fattoriale (int n)
{
        if (n<=1)
                return 1;
        else
                return n * fattoriale (n-1);
}
```

Costruzione di nuovi Tipi di Dati
Abstract Data Type (ADT)

Esempio: il tipo di dati astratto **Libro**

- **Specifica**:
  - Definizione del dominio dei valori:

    Insieme delle quadruple (autore, titolo, editore, anno) dove autore, titolo e editore sono stringhe e anno è un intero

  - Definizione dell'insieme degli operatori:
    - creaLibro (String, String, String, Intero) ✉ libro
    - titolo (libro) ✉ String
    - autore (libro) ✉ String
    - editore (libro) ✉ String
    - anno (libro) ✉ Intero

Costruzione di nuovi Tipi di Dati
Implementazione di ADT

Una possibile implementazione per il tipo di dati libro :

```
struct Libro {
        char autore[26];
        char titolo[53];
        char editore[26];
        int anno;
};

typedef struct Libro *libro;

libro creaLibro (char *A, char *T, char *E, int anno)
{
        libro L;
        L = malloc(sizeof(struct Libro));
        if (!L)   return NULL;
        strcpy(L->autore, A);
        strcpy(L->titolo, T);
        strcpy(L->editore, E);
        L.anno = anno;
        return L;
}
```

```
char *autore (libro L)
{
        char *aut;
        aut = calloc (26, sizeof(char));
        if (aut)
       strcpy(aut, L->autore);
        return aut;
}

char *titolo (libro L)
{
        char *tit;
        tit = calloc (53, sizeof(char));
        if (tit)
      strcpy(tit, L->titolo);
        return tit;
}
```

Costruzione di nuovi Tipi di Dati

Una possibile implementazione per il tipo di dati Libro :

```
char *editore (libro L)
{
        char *ed;
        ed = calloc (26, sizeof(char));
        if (ed)
    strcpy(ed, L.editore);
        return ed;
}

int  anno (libro L)
{
        return L->anno;
}
```

Fine lezione introduttiva