

## I Tipi di Dato Astratto (TDA)

Stefano Mizzaro

Dipartimento di matematica e informatica  
Università di Udine  
<http://www.dimi.uniud.it/mizzaro>  
mizzaro@dimi.uniud.it  
Programmazione, lezione 13  
15 novembre 2006

## Oggi

- Comunicazioni di servizio
- Riassunto
  - Programmazione strutturata
- TDA (Tipo di dato astratto)
  - Evoluzione della programmazione strutturata
  - Primi concetti
  - Occultamento delle informazioni
  - Esempi

Stefano Mizzaro - TDA

2

## Prossime lezioni

- Con me
- 4-6 lezioni
- Poi pausa natalizia (e provetta...)
- Poi a gennaio ultime 6-8 lezioni
  - 12 lezioni, 24 h in totale
- Ricevimento: Martedì 10:00 – 12:00

Stefano Mizzaro - TDA

3

## Esercitazione di auto-valutazione collettiva

- Alcuni si sono iscritti solo al corso e non all'esercitazione
  - Uno è addirittura riuscito a iscriversi all'esercitazione e non al corso (pensavamo fosse impossibile...)
- Alcuni hanno tentato di iscriversi dopo mezzanotte. Rifatelo (abbiamo spostato la scadenza a oggi)
- Alcuni non hanno inserito il loro indirizzo email. Fatelo
- Pochi hanno letto la pagina di istruzioni :-)
- Non bisogna usare Refresh/Back
- C'era un blocco sulla porta 8078 dalla rete WiFi di ateneo. Non c'è più

Stefano Mizzaro - TDA

4

## Scaletta

- Comunicazioni di servizio
- Riassunto
  - Programmazione strutturata
- TDA (Tipo di dato astratto)
  - Evoluzione della programmazione strutturata
  - Occultamento delle informazioni
  - Esempi

Stefano Mizzaro - TDA

5

## Programma del corso

1. La programmazione strutturata (24h)
2. Tipi di dati astratti e occultamento delle informazioni (4h)
3. I fondamenti della programmazione orientata agli oggetti (8h)
4. Le API del Java (10h)
5. Cenni alle teorie della computabilità e complessità (2h)

Stefano Mizzaro - TDA

6

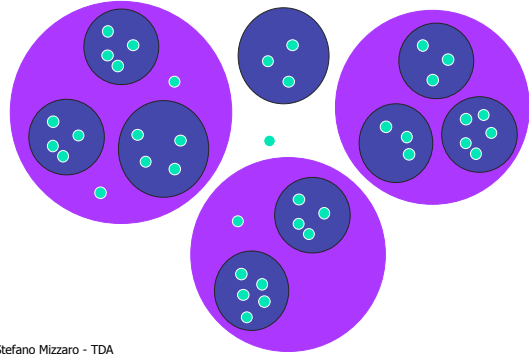
## 1. Programmazione strutturata

- "Mattoni"
- Strutture di controllo
  - Sequenza, Selezione, Iterazione
- Cicli annidati
- Array (unidimensionali, multidimensionali)
- Sottoprogrammi (metodi)
  - Parametri formali e attuali
- Ricorsione

Stefano Mizzaro - TDA

7

## Analisi sistematica: dai mattoni più piccoli alle componenti più grandi



Stefano Mizzaro - TDA

8

## Altro...

- Pila dei record di attivazione, heap
- Attività del programmatore
  - Sviluppo incrementale
  - Ciclo editing-compilazione-esecuzione
  - Testing
- "Bast-Che-Funz"
  - chi se ne frega di come è scritto un programma, basta che funzioni...
  - SE LA PENSATE COSI', NON PASSATE L'ESAME!
  - DOVETE imparare a scrivere programmi di buona qualità
- Anche chi pensa di saper già programmare...

Stefano Mizzaro - TDA

9

## Oggi

- Tipi di Dato Astratto (TDA)
  - Primi concetti
  - Definizione di un TDA
  - Uso di un TDA
  - Principio dell'Occultamento delle informazioni
- Introduzione, ne parleremo anche nella prossima lezione

Stefano Mizzaro - TDA

10

## TDA

- Tipo di Dato Astratto (TDA)
- Abstract Data Type (ADT)
- I metodi (procedure e funzioni) possono essere visti come un modo per estendere il linguaggio di programmazione
  - appena definito un nuovo metodo, è come se il linguaggio avesse acquistato una nuova istruzione
  - È un meccanismo di astrazione
- Ne esiste un altro...
  - Migliore (vedremo perché)

Stefano Mizzaro - TDA

11

## Struttura programma Java?

```

class ... {
    static <tipo> <id> (<parametri>) {
    }
    ...
    static <tipo> <id> (<parametri>) {
    }
    ...
    public static void main (String[] args) {
    }
    ...
    static <tipo> <id> (<parametri>) {
    }
    ...
    static <tipo> <id> (<parametri>) {
    }
    ...
}
  
```

Stefano Mizzaro - TDA

12

## Un altro meccanismo di astrazione: i TDA

- Possiamo estendere il linguaggio definendo nuovi tipi di dato
- Per farlo bisogna:
  - definire "come sono fatti"
  - decidere quali operazioni sono lecite su di essi
- (Tipo = valori + operazioni)

Stefano Mizzaro - TDA

13

## Definire TDA

- Per definire un nuovo tipo di dato si deve specificare una classe
  - il nome della classe rappresenterà il nuovo tipo
  - una classe definisce le sue variabili e i suoi metodi

Stefano Mizzaro - TDA

14

## Definire TDA: la sintassi

```
class <NomeDelTDA> {
    <tipo1> <var1>;
    <tipo2> <var2>;
    ...

    static <tipoMetodo1> <nomeMetodo1>(<parametri1>) {
        ...
    }
    static <tipoMetodo2> <nomeMetodo2>(<parametri2>) {
        ...
    }
    ...
}
```

Stefano Mizzaro - TDA

15

## Convenzione sugli identificatori



- Gli identificatori Java vanno scritti
  - con l'iniziale maiuscola se sono nomi di classi
  - con l'iniziale minuscola se sono nomi di variabili o di metodi
  - se l'identificatore è formato da più parole, queste vanno scritte attaccate e ogni parola deve iniziare con la maiuscola
- Esempi:
  - `Classe1`, `variabile1`, `metodo1`, `AltroNomeDiClasse`, `altroNomeDiVariabile`, `altroNomeDiMetodo`, `nome sbagliato`

Stefano Mizzaro - TDA

16

## Esempio 1: classe coordinate

- Definire un nuovo tipo che rappresenta le coordinate di un punto del piano cartesiano
  - Decidiamo il nome della classe: `Coordinate`
  - Decidiamo quali informazioni/dati saranno presenti nella classe: `x`, `y`

```
class Coordinate {
    int x;
    int y;
}
```

oppure

```
class Coordinate {
    double x;
    double y;
}
```

Stefano Mizzaro - TDA

17

## Esempio 2: classe ora

- Definire un nuovo tipo che rappresenta un'ora del giorno
  - Decidiamo il nome della classe: `Ora`
  - Decidiamo quali informazioni/dati saranno presenti nella classe:
    - ore, minuti, secondi oppure
    - secondi trascorsi dalla mezzanotte

```
class Ora {
    int ore;
    int minuti;
    int secondi;
}
```

oppure

```
class Ora {
    int secondi;
}
```

Stefano Mizzaro - TDA

18

### Esempio 3: classe freccia orizzontale

- Definire un nuovo tipo che rappresenta una freccia orizzontale
  - Decidiamo il nome della classe: **FrecciaOrizzontale**
  - Decidiamo quali informazioni/dati saranno presenti nella classe: lunghezza, verso

```
class FrecciaOrizzontale {
    int lunghezza;
    char verso;
}
```

oppure

```
class FrecciaOrizzontale {
    int lunghezza;
    boolean versoDestra;
}
```

Stefano Mizzaro - TDA 19

### Usare TDA: la notazione puntata

- Una volta definita una classe abbiamo un nuovo tipo
  - (non abbiamo scritto un programma; non c'è un main)
- Ora si possono:
  - dichiarare variabili di quella classe (di quel tipo)
  - creare nuovi valori (istanze)
  - usare le variabili
- (in un'altra classe/file... ad es. nel main... vedremo)

Stefano Mizzaro - TDA 20

### Dichiarazione

- Analogamente a come si dichiarano le variabili con tipo primitivo:
  - `int x, k, count;`
  - `Coordinate punto, coppia, pluto;`
  - Ora `oraDelGiorno, pippo;`
- Dichiarare una variabile non basta. Per usare un TDA bisogna creare esplicitamente un valore
  - Bisogna allocare dello spazio in memoria
- Lo spazio per i tipi primitivi invece viene allocato in modo automatico

Stefano Mizzaro - TDA 21

### L'istruzione new

- Dichiarazione e istanziazione/allocazione:
  - `Coordinate punto;`
  - `punto = new Coordinate();`
- oppure, in una sola riga:
  - `Coordinate punto = new Coordinate();`
- new**
  - Alloca lo spazio nello heap per una nuova istanza
  - Viene creato un nuovo oggetto che verrà riferito dalla variabile
  - Ricorda qualcosa?

Stefano Mizzaro - TDA 22

### Istanziamento/allocazione

Dichiarazione: `Coordinate punto;`      Istanziamento: `punto = new Coordinate();`

Pila dei record di attivazione      Heap

Stefano Mizzaro - TDA 23

### La notazione puntata

- Una volta dichiarata una variabile e istanziato un TDA, si può accedere ai suoi attributi, ovvero alle variabili, usando questa sintassi:
  - `variabileTDA.variabileComponente`
- I metodi dei TDA invece si invocano premettendo il nome della classe:
  - `TDA.metodo(parametri)`
- Esempi
  - `punto.x = 2;`
  - `punto.y = 3 + punto.x;`
  - `System.out.println(punto.y);`

Stefano Mizzaro - TDA 24

## Esempio 1

- Scrivere un programma che
  - crea due punti le cui coordinate sono lette da tastiera usando la classe `Leggi` e
  - stampa la distanza tra i due punti
- Per calcolare la radice quadrata usate la funzione `Math.sqrt`

Stefano Mizzaro - TDA

25

```

/* Definizione del nuovo tipo da usare */
class Coordinate {
    int x; //primo componente
    int y; //secondo componente
}

/* Definizione del programma */
class Distanza {
    public static void main(String[] args) {
        //dichiarazione e istanziazione di due punti
        Coordinate punto1 = new Coordinate();
        Coordinate punto2 = new Coordinate();

        System.out.println("Inserisci le prime due coordinate");
        //Leggi.unInt() aspetta un intero da tastiera
        //l'istruzione di assegnazione salva l'intero letto
        //nella variabile x del TDA punto1
        punto1.x = Leggi.unInt();
        //analogamente viene salvato un intero letto da
        //tastiera nella variabile y del TDA punto1
        punto1.y = Leggi.unInt();
        ...
    }
}

```

```

...
System.out.println("Inserisci le altre coordinate");
punto2.x = Leggi.unInt();
punto2.y = Leggi.unInt();

//le variabili di un TDA possono essere usate anche
//nelle espressioni
int distanzaX = punto1.x - punto2.x;
int distanzaY = punto1.y - punto2.y;

double distanza = Math.sqrt(distanzaX*distanzaX +
                             distanzaY*distanzaY);
System.out.println("I due punti inseriti distano " +
                  distanza);
}
}

```

Stefano Mizzaro - TDA

27

## Esempio 2

- Scrivere un programma che legge da tastiera un'ora del giorno e la stampa

```

...
Ora x = new Ora();
//Legge un'ora del giorno
x.ore = Leggi.unInt();
x.minuti = Leggi.unInt();
x.secondi = Leggi.unInt();

System.out.println("Sono le "
                  +x.ore+" "+x.minuti+" "+x.secondi);
...

```

Stefano Mizzaro - TDA

28

## Esempio 2

- Con l'altra definizione della classe `Ora`

```

...
Ora x = new Ora();
//Legge un'ora del giorno
int ore = Leggi.unInt();
int minuti = Leggi.unInt();
int secondi = Leggi.unInt();

x.secondi = ore*3600+minuti*60+secondi;

System.out.println("Ora sono le "+(x.secondi/3600)+":"+
                  +(x.secondi%3600)/60+" "+
                  +(x.secondi%60));
...

```

Stefano Mizzaro - TDA

29

## Esempio 3

- Scrivere un programma che legge da tastiera la lunghezza e il verso di una freccia e poi la stampa

```

...
FrecciaOrizzontale x =
    new FrecciaOrizzontale();
//Legge lunghezza e verso
x.lunghezza = Leggi.unInt();
x.verso = Leggi.unChar();

if (x.verso == 's')
    System.out.print("<");
for (int i=0; i<x.lunghezza; i++)
    System.out.print("-");
if (x.verso == 'd')
    System.out.println(">");
else
    System.out.println();
...

```

Stefano Mizzaro - TDA

30

### Esempio 3

- Oppure, con l'altra definizione della classe **FrecciaOrizzontale**

```

...
FrecciaOrizzontale x =
    new FrecciaOrizzontale();
//Legge lunghezza e verso
x.lunghezza = Leggi.unInt();
char verso = Leggi.unChar();
x.versoDestra = (verso == 'd');

if (!x.versoDestra)
    System.out.print("<");
for (int i=0; i<x.lunghezza; i++)
    System.out.print("-");
if (x.versoDestra)
    System.out.println(">");
else
    System.out.println();
...

```

Stefano Mizzaro - TDA

31

### L'occultamento delle informazioni

- Negli esempi precedenti non è evidente il vantaggio di usare i TDA
  - L'astrazione è ancora limitata
  - Per usare un TDA si fa riferimento direttamente a come è fatto (alla sua implementazione)
  - Mancano le operazioni del tipo
- Negli esempi 2 e 3, cambiando la definizione della classe dovevamo cambiare anche il codice che usava quei TDA
  - (e anche nell'es. 1, se avessimo avuto definizioni diverse)

Stefano Mizzaro - TDA

32

### L'occultamento delle informazioni

- È buona norma nascondere l'implementazione dei TDA
- L'accesso e la modifica delle variabili interne avviene solo tramite metodi
  - (le operazioni del tipo)
- In questo modo si realizza un'astrazione maggiore:
  - chi userà la classe dovrà conoscere solo l'insieme dei metodi (la sua interfaccia)
  - non "come è fatto dentro" (l'implementazione)
- Tipo di dato **astratto**: si astrae dall'implementazione (quando si usa)

Stefano Mizzaro - TDA

33

### I modificatori

- In Java si possono nascondere le variabili di una classe usando il modificatore **private**
- Esempio:

```

class Coordinate {
    private int x;
    private int y;
}

```

Stefano Mizzaro - TDA

34

### Variabili private

- Se una variabile è dichiarata privata, può essere riferita solo nei metodi della sua classe
- Se un metodo di un'altra classe prova ad accedere ad una variabile privata, il compilatore segnala un errore
  - Se si nascondono le variabili di una classe, si devono aggiungere dei metodi per modificarle
  - Le operazioni del tipo

Stefano Mizzaro - TDA

35

### Esempio 1

```

class Coordinate {
    private int x;
    private int y;
    static void setCoordinate(Coordinate p,
                              int a, int b) {
        p.x = a;
        p.y = b;
    }
    static int primaCoordinata(Coordinate a) {
        return a.x;
    }
    static int secondaCoordinata(Coordinate a) {
        return a.y;
    }
}

```

Stefano Mizzaro - TDA

36

## Esempio 2

```
class Ora {
    private int ore, minuti, secondi;
    static void impostaOra(Ora x, int o, int m, int s) {
        x.ore = o;
        x.minuti = m;
        x.secondi = s;
    }
    static int getOre(Ora x) {return x.ore;}
    static int getMinuti(Ora x) {return x.minuti;}
    static int getSecondi(Ora x) {return x.secondi;}
}
```

Stefano Mizzaro - TDA

37

## Esempio 2: altra implementazione

```
class Ora {
    private int secondi;
    static void impostaOra(Ora x, int o, int m, int s) {
        x.secondi = o*3600+m*60+s;
    }
    static int getOre(Ora x) {return x.secondi/3600;}
    static int getMinuti(Ora x) {
        return (x.secondi%3600)/60;
    }
    static int getSecondi(Ora x) {return x.secondi%60;}
}
```

Stefano Mizzaro - TDA

38

## Esempio 2: vantaggi dell'occultamento/astrazione

```
...
Ora h = new Ora();
Ora.impostaOra(h, 12, 22, 31);
System.out.println("Sono le ore "
    +Ora.getOre(h));
...
```

- Questo codice va bene con entrambe le implementazioni!
- Stiamo astruendo dall'implementazione

Stefano Mizzaro - TDA

39

## Riassunto

- Riassunto programmazione strutturata
- TDA
  - Altro meccanismo di astrazione
  - Definizione (**class**)
  - Uso: dichiarazione e istanziazione/allocazione (**new**); notazione puntata)
  - Occultamento delle informazioni, astrazione dall'implementazione (**private** e metodi)
  - Esempi
- Prossima lezione
  - Fine TDA: riassunto, sistematizzazione, aggiunte e puntualizzazioni

Stefano Mizzaro - TDA

40