TIPOLOGIE DI LINGUAGGI DI PROGRAMMAZIONE ESISTENTI

Giovanni Scavello

Classe di concorso A041

Introduzione

- Contestualizzazione
- Didattica e metodologia
- Contenuti Specifici della lezione
- Conclusioni

Contestualizzazione

Secondo biennio	
Conoscenze	***Abilità***
Relazioni fondamentali tra macchine, problemi, informazioni e linguaggi Linguaggi e macchine a vari livelli di astrazione. Paradigmi di programmazione. Logica iterativa e ricorsiva.	Progettare e implementare algoritmi utilizzando diverse strutture di dati. Analizzare e confrontare algoritmi diversi per la soluzione dello stesso problema.
Principali strutture dati e loro implementazione. File di testo. Teoria della complessità algoritmica. Programmazione ad oggetti. Programmazione guidata dagli eventi e interfacce grafiche. Strumenti per lo sviluppo del software e supporti per la robustezza dei programmi. Linguaggi per la definizione delle pagine web. Linguaggio di programmazione lato client per la gestione locale di eventi in pagine web. Lessico e terminologia tecnica di settore anche in lingua inglese. Normative di settore nazionale e comunitaria sulla sicurezza .	Scegliere il tipo di organizzazione dei dati più adatto a gestire le informazioni in una situazione data. Gestire file di testo. Progettare e implementare applicazioni secondo il paradigma ad oggetti. Progettare e realizzare interfacce utente. Progettare, e realizzare e gestire pagine web statiche con interazione locale. Utilizzare il lessico e la terminologia tecnica di settore anche in lingua inglese. Applicare le normative di settore sulla sicurezza.

- Curriculum
  - ✔ Secondo biennio
  - ✔ Istituto Tecnico Informatica e Telecomunicazioni
  - ✔ Terzo anno
  - ✔ Informatica
- Composizione classe
  - ✔ 23 studenti
  - ✔ 1 BES con lieve dislessia

Contestualizzazione

UDA

Introduzione alla programmazione e agli algoritmi (60 ore)

- Introduzione agli algoritmi (6 ore)
- Introduzione ai linguaggi di programmazione (2 ore)
- Programmazione strutturata:
  - Concetto di variabile
  - sequenza, ciclo e condizione (36 ore)
    - Pseudocodice
    - Diagrammi di flusso
- Introduzione al linguaggio C (10 ore)
- Attività di recupero (6 ore)

Contestualizzazione

UDA

Introduzione alla programmazione e agli algoritmi (60 ore)

- Introduzione agli algoritmi (6 ore)
- Introduzione ai linguaggi di programmazione (2 ore)
- Programmazione strutturata:
  - Concetto di variabile
  - sequenza, ciclo e condizione (36 ore)
    - Pseudocodice
    - Diagrammi di flusso
- Introduzione al linguaggio C (10 ore)
- Attività di recupero (6 ore)

PREREQUISITI

- Conoscenze di base sull'architettura dei calcolatori
- Conoscenze di base sui sistemi operativi, in particolare sul concetto di shell
- Conoscenza dell'ambiente Scratch

Contestualizzazione

UDA

Introduzione alla programmazione e agli algoritmi (60 ore)

- Introduzione agli algoritmi (6 ore)
- Introduzione ai linguaggi di programmazione (2 ore)
- Programmazione strutturata:
  - Concetto di variabile
  - sequenza, ciclo e condizione (36 ore)
  - Pseudocodice
  - Diagrammi di flusso
- Introduzione al linguaggio C (10 ore)
- Attività di recupero (6 ore)

PREREQUISITI

- Conoscenze di base sull'architettura dei calcolatori
- Conoscenze di base sui sistemi operativi, in particolare sul concetto di shell
- Conoscenza dell'ambiente Scratch

OBIETTIVI

- Elementari
- Conoscere la definizione di algoritmo
- Conoscere la definizione di linguaggio di programmazione
- Conoscere i diversi tipi di linguaggi
- Conoscere le strutture della programmazione
- Convergenti
- Descrivere un algoritmo in linguaggio naturale
- Rappresentare un algoritmo in pseudocodice/diagramma a blocchi
- Tradurre un diagramma a blocchi in C
- Divergenti
- Formulare un algoritmo per la risoluzione di un problema

Contestualizzazione

UDA

Introduzione alla programmazione e agli algoritmi (60 ore)

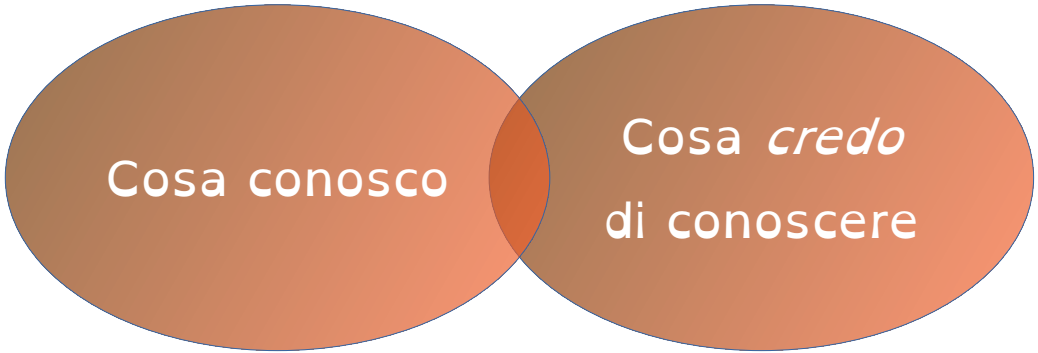
- Introduzione agli algoritmi (6 ore)
- **Introduzione ai linguaggi di programmazione (2 ore)**
- Programmazione strutturata:
  - Concetto di variabile
  - sequenza, ciclo e condizione (36 ore)
  - Pseudocodice
  - Diagrammi di flusso
- Introduzione al linguaggio C (10 ore)
- Attività di recupero (6 ore)

PREREQUISITI

- Conoscenze di base sull'architettura dei calcolatori
- Conoscenze di base sui sistemi operativi, in particolare sul concetto di shell
- Conoscenza dell'ambiente Scratch

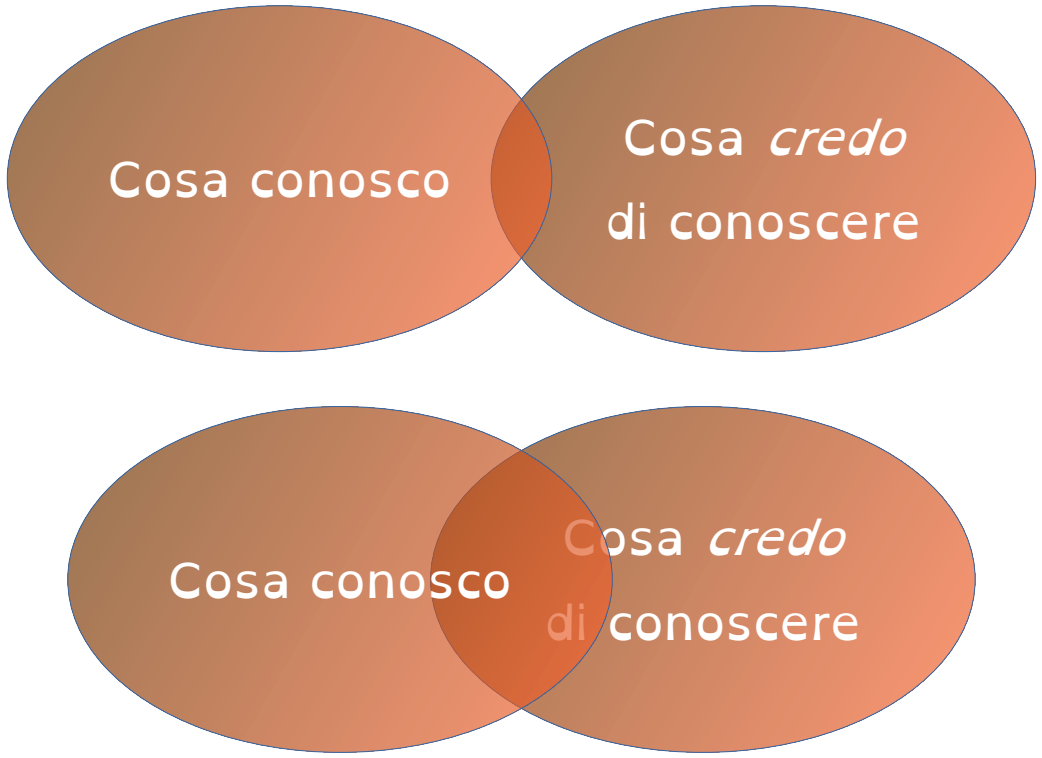
OBIETTIVI

- Elementari
- Conoscere la definizione di algoritmo
- Conoscere la definizione di linguaggio di programmazione
- Conoscere i diversi tipi di linguaggi
- Conoscere le strutture della programmazione
- Convergenti
- Descrivere un algoritmo in linguaggio naturale
- Rappresentare un algoritmo in pseudocodice/diagramma a blocchi
- Tradurre un diagramma a blocchi in C
- Divergenti
- Formulare un algoritmo per la risoluzione di un problema



Searching for Explanations: How the Internet Inflates Estimates of Internal Knowledge [Matthew Fisher, Mariel K. Goddu, and Frank C. Keil] Journal of Experimental Psychology, Online First Publication, March 30, 2015. http://dx.doi.org/10.1037/xge0000070

Didattica e metodologia



Searching for Explanations: How the Internet Inflates Estimates of Internal Knowledge [Matthew Fisher, Mariel K. Goddu, and Frank C. Keil] Journal of Experimental Psychology, Online First Publication, March 30, 2015. http://dx.doi.org/10.1037/xge0000070


Consapevolezza

Didattica e metodologia

Didattica e metodologia

Contenuti specifici della lezione

- Argomenti trattati
  - Problemi, soluzioni, rappresentazioni
  - Comunicare con le macchine
  - Livelli di astrazione nei linguaggi
  - Interpretazione e compilazione
- Modalità
  - Lezione partecipata/dialogata + attività di laboratorio
- Tempistica
  - (30 minuti + 1:30)

Contenuti Specifici della lezione

Contenuti Specifici della lezione

Obiettivi della lezione

Saper definire un linguaggio

Conoscere i diversi tipi di linguaggio

Saper individuare il linguaggio adatto a risolvere un problema

Contenuti Specifici della lezione

Obiettivi della lezione

Saper definire un linguaggio

Conoscere i diversi tipi di linguaggio

Saper individuare il linguaggio adatto a risolvere un problema

Attrattori

Terminale del sistema operativo

Mbot/mBlock

web

Contenuti Specifici della lezione

Obiettivi della lezione

Saper definire un linguaggio

Conoscere i diversi tipi di linguaggio

Saper individuare il linguaggio adatto a risolvere un problema

Attrattori

Terminale del sistema operativo

Mbot/mBlock

web

Attività proposte

Utilizzo BASH

Utilizzo di ambiente di programmazione visuale/strutturata

Contenuti Specifici della lezione

Obiettivi della lezione

Saper definire un linguaggio

Conoscere i diversi tipi di linguaggio

Saper individuare il linguaggio adatto a risolvere un problema

Attrattori

Terminale del sistema operativo

Mbot/mBlock

web

Attività proposte

Utilizzo BASH

Utilizzo di ambiente di programmazione visuale/strutturata

Verifica formativa

Linguaggio naturale

???

Sequenza di impulsi elettrici

Linguaggio naturale

Analisi del problema e scelta della strategia

Schema risolutivo

Sequenza di impulsi elettrici

Linguaggio naturale

Analisi del problema e scelta della strategia

Schema risolutivo

Specificità dell'informatica

Linguaggio di programmazione

Sequenza di impulsi elettrici

Linguaggio naturale

Analisi del problema e scelta della strategia

Schema risolutivo

Specificità dell'informatica

Linguaggio di programmazione

Compilatori

Interpreti

Codice Binario

Sequenza di impulsi elettrici

Linguaggio naturale

Analisi del problema e scelta della strategia

Schema risolutivo

Specificità dell'informatica

Linguaggio di programmazione

Compilatori

Interpreti

Codice Binario

Dispositivi elettronici

Sequenza di impulsi elettrici

Fissiamo le idee

Definizioni su https://en.wikipedia.org/wiki/Programming_language

Fissiamo le idee

Cos'è un linguaggio di programmazione

Definizioni su https://en.wikipedia.org/wiki/Programming_language

Fissiamo le idee

A programming language is a formal computer language or constructed language designed to communicate instructions to a machine, particularly a computer. Programming languages can be used to create programs to control the behavior of a machine or to express algorithms.

Cos'è un linguaggio di programmazione

Definizioni su https://en.wikipedia.org/wiki/Programming_language

Fissiamo le idee

A programming language is a formal computer language or constructed language designed to communicate instructions to a machine, particularly a computer. Programming languages can be used to create programs to control the behavior of a machine or to express algorithms.

Cos'è un linguaggio di programmazione

A formal language is a set of strings of symbols that may be constrained by rules that are specific to it.

Cos'è un linguaggio formale

Definizioni su https://en.wikipedia.org/wiki/Programming_language

Fissiamo le idee

A programming language is a formal computer language or constructed language designed to communicate instructions to a machine, particularly a computer. Programming languages can be used to create programs to control the behavior of a machine or to express algorithms.

Cos'è un linguaggio di programmazione

A formal language is a set of strings of symbols that may be constrained by rules that are specific to it.

Cos'è un linguaggio formale

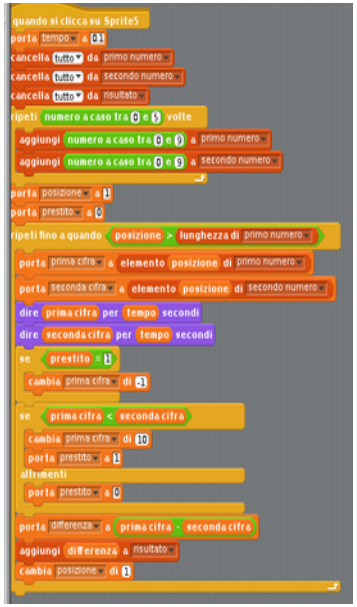
In mathematical linguistics and the theory of automata (cf. Automata, theory of) one considers various effective ways of specifying a formal language, principally by means of formal grammars (cf. Grammar, formal)

Cosa definisce un linguaggio

Definizioni su https://en.wikipedia.org/wiki/Programming_language

Fissiamo le idee

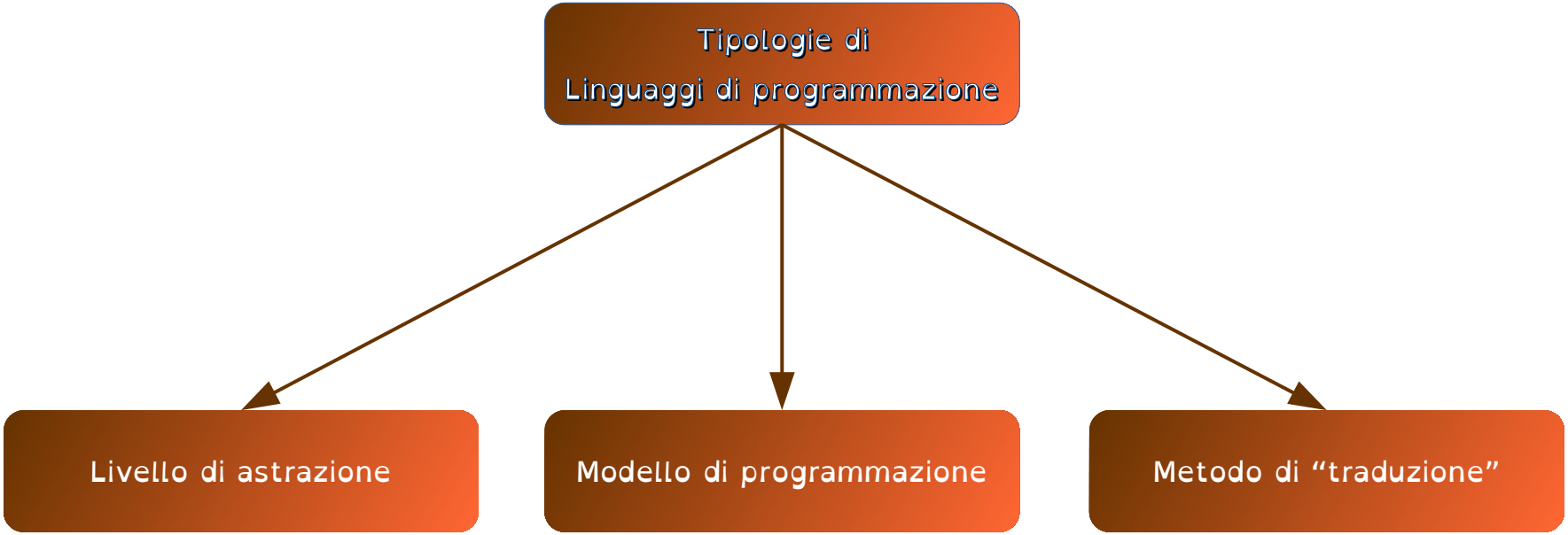
- Sorgente
  - Testo o schema scritto in un linguaggio di programmazione comprensibile da una persona
- Eseguibile
  - Sequenza di codice binario che rappresenta i segnali di attivazione dei dispositivi interni

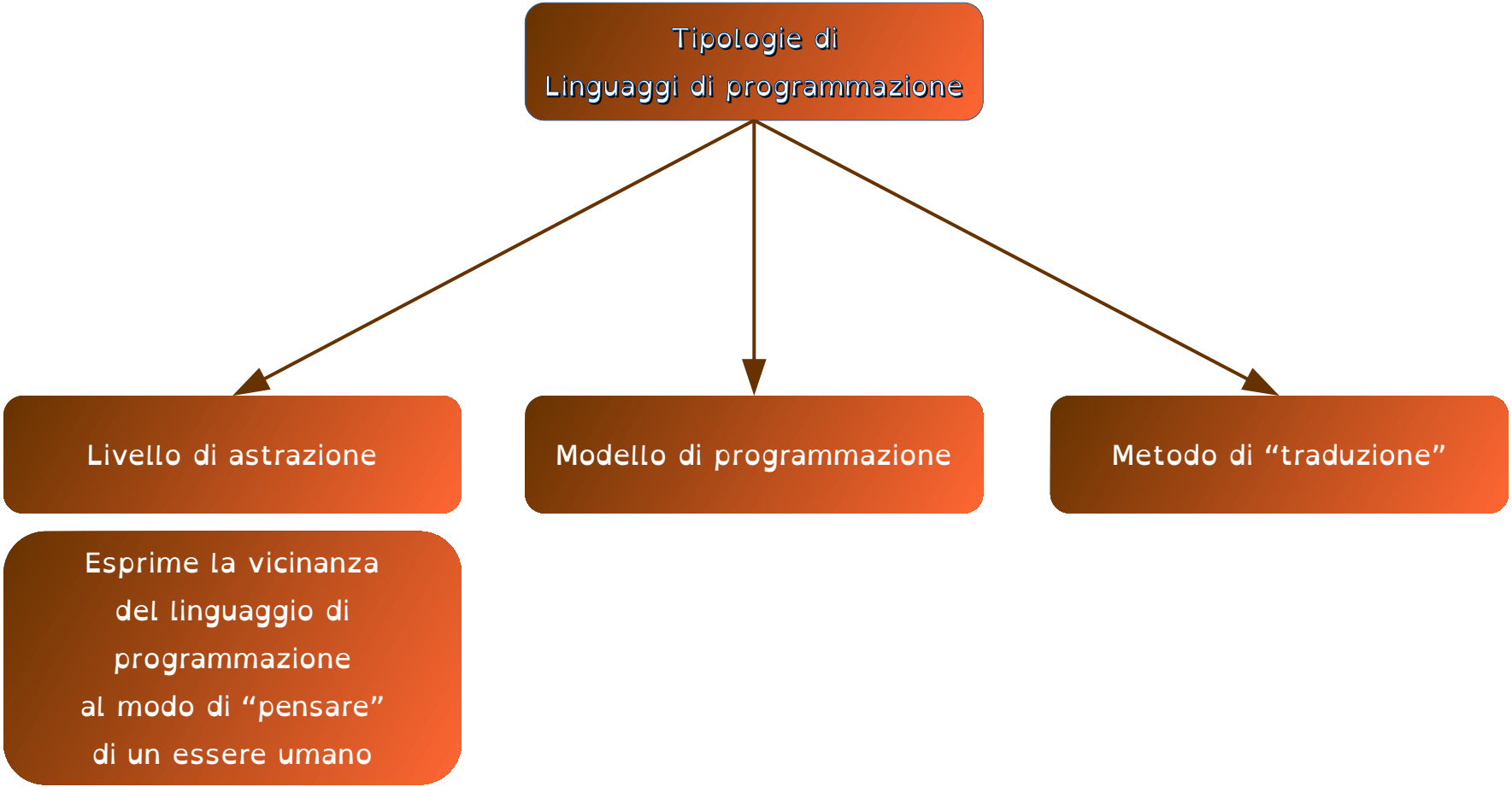


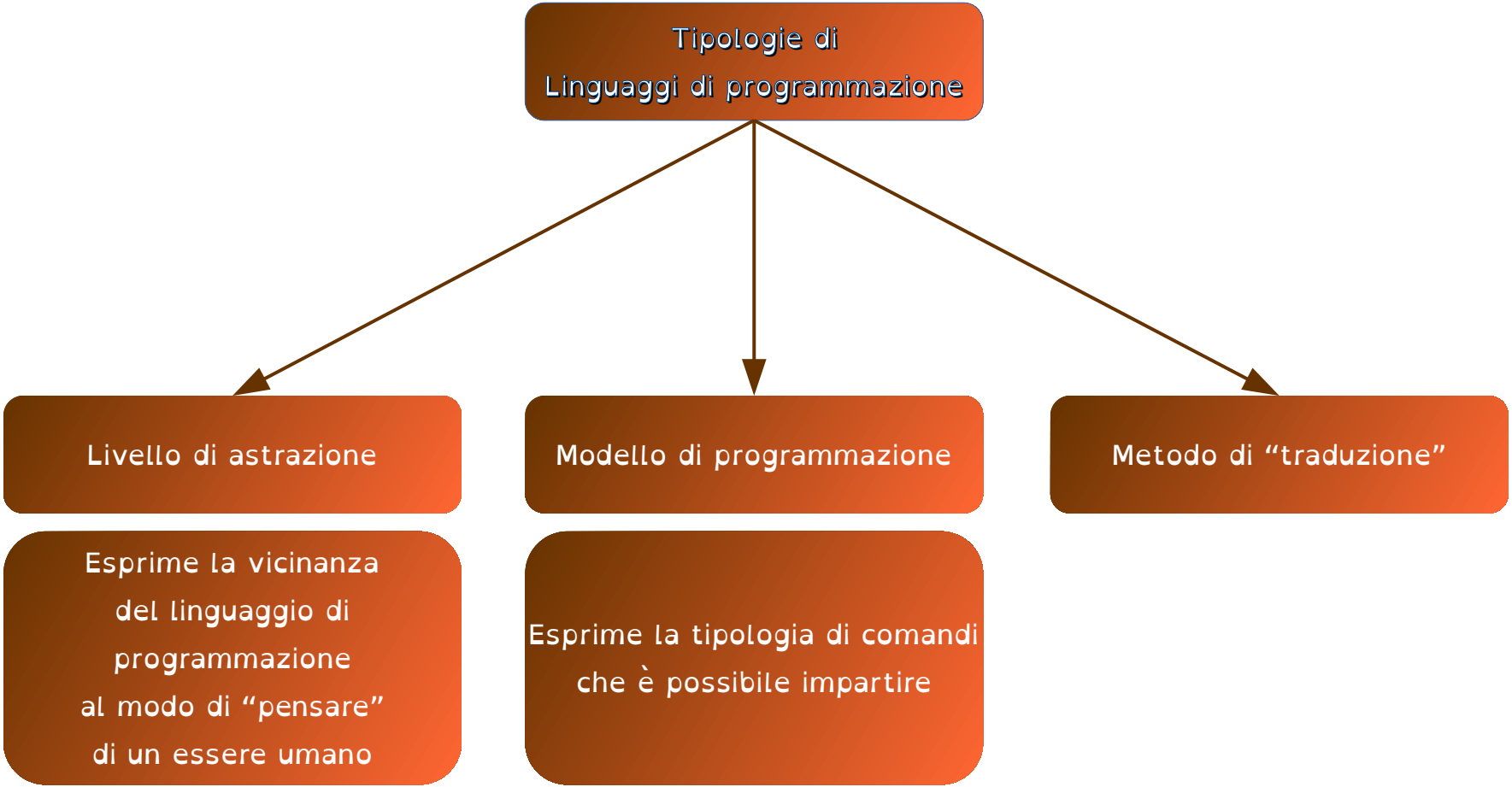

```
nuova variabile lista
nuova variabile numero
nuova variabile posizione
ripeti 10 volte
    chiedi "inserisci un numero" e attendi
    aggiungi risposta a lista
porta numero a elemento 1 di lista
porta posizione a 2
ripeti fino a quando posizione > 10
    se numero > elemento posizione di lista
        porta numero a elemento posizione di lista
    porta posizione a posizione + 1
dire "numero è"
dire numero
```

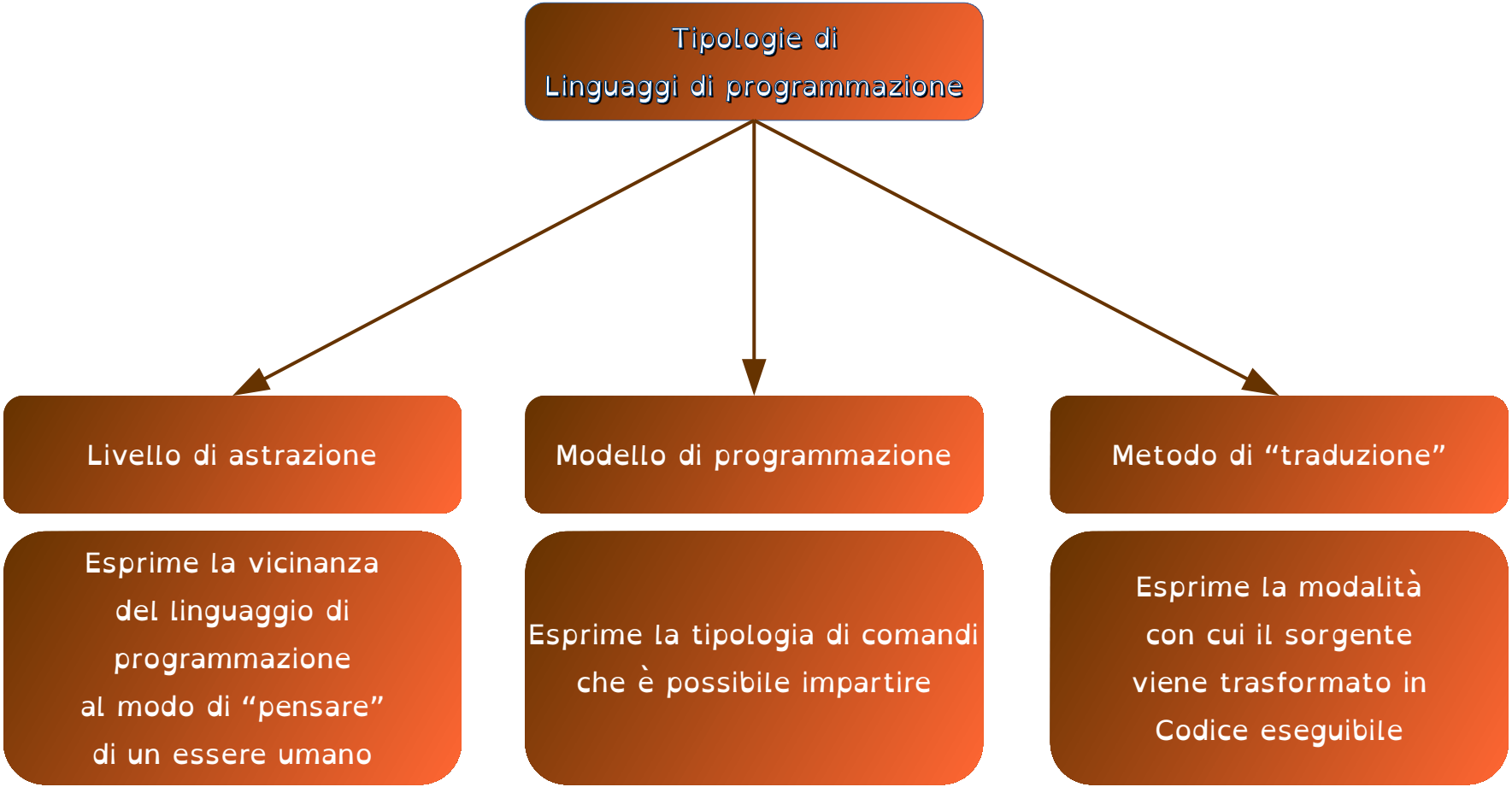
```
30 02 00 00  00 00 00 00  30 02 00 00  00 00 00 00
08 00 00 00  00 00 00 00  04 00 00 00  04 00 00 00
8C 02 00 00  00 00 00 00  8C 02 40 00  00 00 00 00
8C 02 40 00  00 00 00 00  44 00 00 00  00 00 00 00
44 00 00 00  00 00 00 00  04 00 00 00  00 00 00 00
07 00 00 00  04 00 00 00  70 8C 01 00  00 00 00 00
70 8C 61 00  00 00 00 00  70 8C 61 00  00 00 00 00
00 00 00 00  00 00 00 00  08 00 00 00  00 00 00 00
08 00 00 00  00 00 00 00  50 E5 74 64  04 00 00 00
AC 46 01 00  00 00 00 00  AC 46 41 00  00 00 00 00
AC 46 41 00  00 00 00 00  6C 0A 00 00  00 00 00 00
6C 0A 00 00  00 00 00 00  04 00 00 00  00 00 00 00
51 E5 74 64  06 00 00 00  00 00 00 00  00 00 00 00
00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
10 00 00 00  00 00 00 00  52 E5 74 64  04 00 00 00
70 8C 01 00  00 00 00 00  70 8C 61 00  00 00 00 00
70 8C 61 00  00 00 00 00  90 03 00 00  00 00 00 00
90 03 00 00  00 00 00 00  01 00 00 00  00 00 00 00
2F 6C 69 62  36 34 2F 6C  64 2D 6C 69  6E 75 78 2D
78 38 36 2D  36 34 2E 73  6F 2E 32 00  04 00 00 00
10 00 00 00  01 00 00 00  47 4E 55 00  00 00 00 00
02 00 00 00  06 00 00 00  20 00 00 00  04 00 00 00
```

Tipologie di Linguaggi di programmazione


Tipologie di
Linguaggi di programmazione
Livello di astrazione
Modello di programmazione
Metodo di “traduzione”


Tipologie di
Linguaggi di programmazione
Livello di astrazione
Esprime la vicinanza
del linguaggio di
programmazione
al modo di “pensare”
di un essere umano
Modello di programmazione
Metodo di “traduzione”

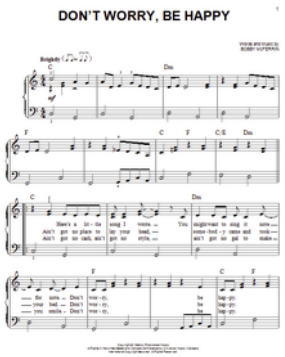

Tipologie di
Linguaggi di programmazione
Livello di astrazione
Modello di programmazione
Metodo di “traduzione”
Esprime la vicinanza
del linguaggio di
programmazione
al modo di “pensare”
di un essere umano
Esprime la tipologia di comandi
che è possibile impartire


Tipologie di
Linguaggi di programmazione
Livello di astrazione
Esprime la vicinanza del linguaggio di programmazione al modo di “pensare” di un essere umano
Modello di programmazione
Esprime la tipologia di comandi che è possibile impartire
Metodo di “traduzione”
Esprime la modalità con cui il sorgente viene trasformato in Codice eseguibile

Livello di astrazione

Livello di astrazione

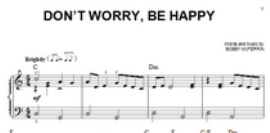
Livello di astrazione



Livello di astrazione

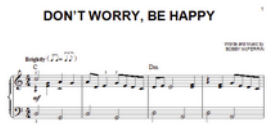
Livello di astrazione

Livello di astrazione



Linguaggi di altissimo livello

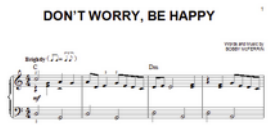
Livello di astrazione



Linguaggi di altissimo livello

Linguaggi di alto livello

Livello di astrazione



Linguaggi di altissimo livello

Linguaggi di alto livello

Linguaggi di basso livello

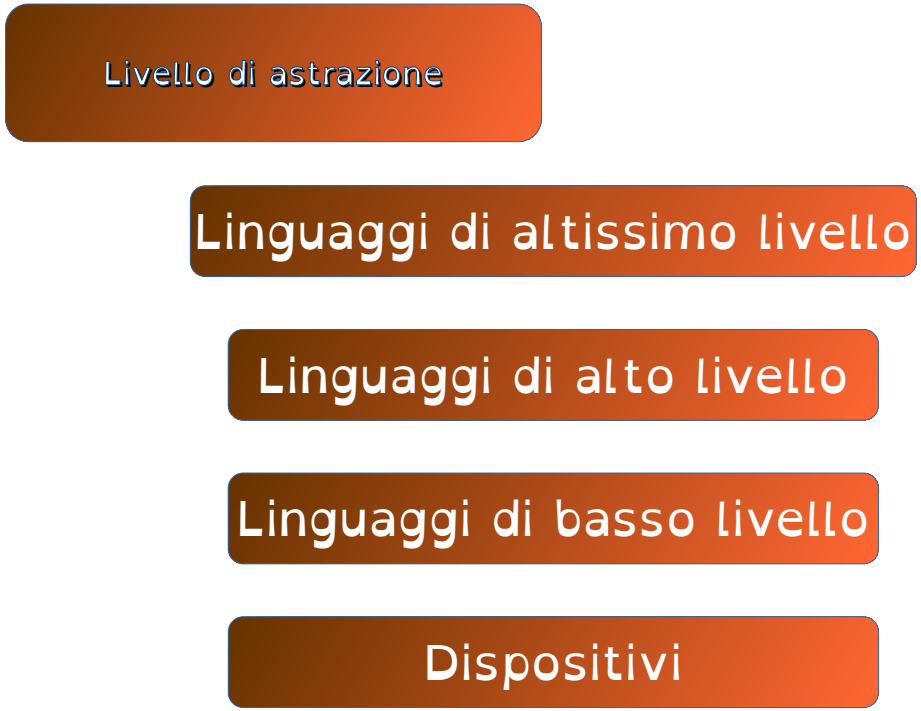
Livello di astrazione

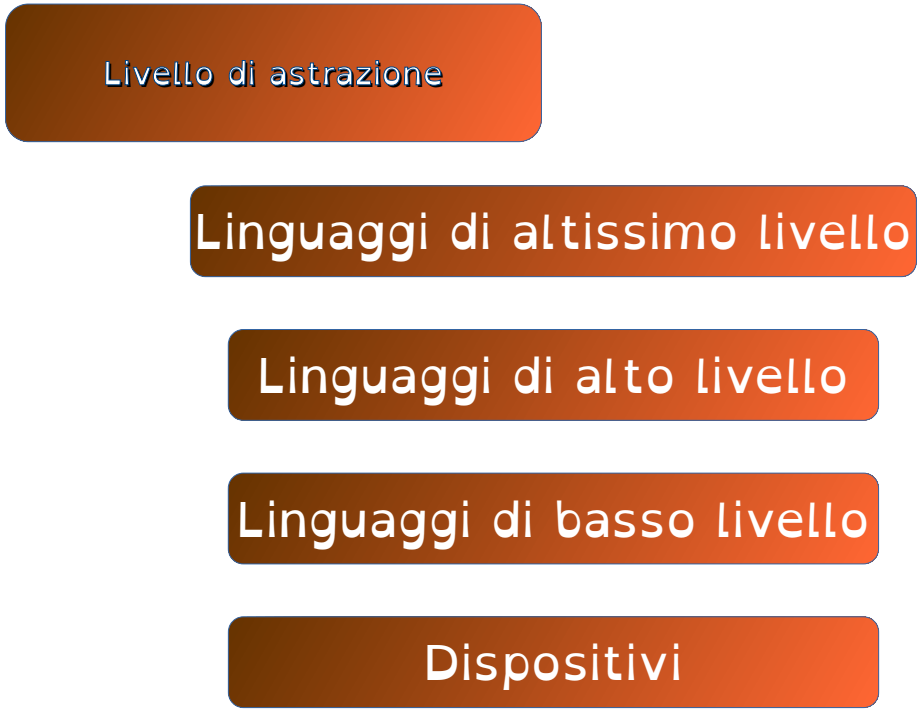
Linguaggi di altissimo livello

Linguaggi di alto livello

Linguaggi di basso livello

Dispositivi


Livello di astrazione
Linguaggi di altissimo livello
Linguaggi di alto livello
Linguaggi di basso livello
Dispositivi



http://rosettacode.org/wiki/Hello_world/Text

Livello di astrazione

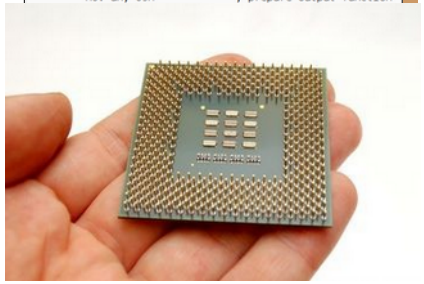
```
SELECT 'Hello world!';
```

```
#include <stdlib.h>
#include <stdio.h>

int main(void)
{
```

```
DOSSEG
.MODEL TINY
.DATA
TXT DB "Hello world!$"
.CODE
START:
        MOV ax, @DATA
        MOV ds, ax

        MOV ah, 09h             ; prepare output function
```



Linguaggi di altissimo livello

Linguaggi di alto livello

Linguaggi di basso livello

Dispositivi

http://rosettacode.org/wiki/Hello_world/Text

Livello di astrazione

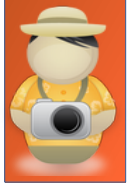
Linguaggi di altissimo livello

Linguaggi di alto livello

Linguaggi di basso livello

Dispositivi

semplicità

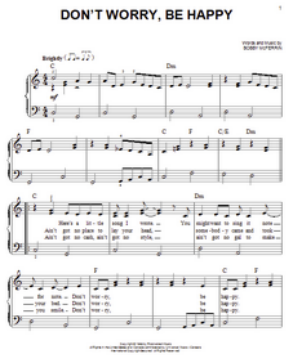


Efficienza

Modello di programmazione

Imperativi

Dichiarativi




```c
#include <stdlib.h>
#include <stdio.h>

int main(void)
{
  printf("Hello world!\n");
  return EXIT_SUCCESS;
}
```

```
DOSSEG
.MODEL TINY
.DATA
TXT DB "Hello world!$"
.CODE
START:
        MOV ax, @DATA
        MOV ds, ax

        MOV ah, 09h             ; prepare output function
        MOV dx, OFFSET TXT      ; set offset
        INT 21h                 ; output string TXT

        MOV AX, 4C00h           ; go back to DOS
        INT 21h
END START
```

```sql
SELECT 'Hello world!';
```

http://www.slideshare.net/dibari.92/linguaggi-di-programmazione-tipi-di-linguaggio-compilatori-ed-interpreti

Metodo di “traduzione

Interpreti

Un interprete, in informatica e nella programmazione, è un programma in grado di eseguire altri programmi a partire direttamente dal relativo codice sorgente.

Ha lo scopo di eseguire un programma in un linguaggio di alto livello, senza la previa compilazione dello stesso (codice oggetto) cioè di eseguire le istruzioni nel linguaggio usato, traducendole di volta in volta in istruzioni in linguaggio macchina.

Compilatori

A differenza di un interprete, un compilatore non esegue il programma che riceve in ingresso, ma lo traduce in linguaggio macchina (memorizzando su file il codice oggetto pronto per l'esecuzione diretta da parte del processore).

Per qualunque linguaggio di programmazione si può scrivere sia un interprete che un compilatore, pertanto le espressioni linguaggio interpretato e linguaggio compilato sono scorrette, essendo interpretazione e compilazione concetti afferenti alla implementazione di un linguaggio e non al linguaggio in sé.

Definizioni su https://it.wikipedia.org/wiki/Interprete_%28informatica%29

L
A
B

- **Esercitazione** Utilizzo della bash in linux
- **Obiettivo** introdurre il concetto di interprete e di sintassi
  - Elenco
  - Elenco e ordinamento
  - Ricerca in file di testo
  - Assegnazione esercizio e consultazione guida on line

L
A
B

- **Esercitazione** Utilizzo dell'ambiente mblock per la movimentazione di un mbot
- **Obiettivo** visualizzare il codice di livello intermedio prodotto dai blocchi utilizzati per la movimentazione

Conclusioni

- La lezione proposta rappresenta un'introduzione molto generale al tema dei linguaggi
- Gli esempi proposti hanno soprattutto l'obiettivo di stimolare la curiosità e creare un gancio per il futuro
  (ex: approfondimento personale sullo scripting bash, programmazione di microcontrollori, etc...)
- Far capire che esistono linguaggi più comodi e linguaggi meno comodi, a seconda di ciò che si vuole fare

Thank you for your attention