# Seconda consegna per l’esame di Web semantico

---

# Sistema per la raccomandazione di materiale didattico per il linguaggio C

**Lorenzo Bacchiani**

Anno 2019-2020

# Indice generale

# Capitolo 1

## Introduzione

L’obiettivo di questa relazione è mostrare sia la struttura di una ontologia scritta attraverso gli standard del Web Semantico come RDF, RDFS e OWL sia mostrarne una sua applicazione pratica. In particolare è stato sviluppato un sistema di raccomandazione per il materiale didattico relativo al dominio del linguaggio di programmazione C. Tale sistema, come verrà descritto più avanti, lavora in due modalità differenti:

- A partire da un file rdf contenente le annotazioni di un file .c, il sistema fornisce il materiale relativo ai concetti contenuti all’interno di tale file e raccomanda il materiale relativo ai concetti strettamente correlati a quelli presenti.
- Simula la creazione di n profili di studenti ognuno contente informazioni come livello di educazione, argomenti di interesse, tipi di materiale preferito (letture, materiale scolastico ecc.). Per ogni profilo creato fornisce in output il materiale corrispondente alle preferenze fornite e suggerendo materiale relativo a concetti correlati agli argomenti di interesse che soddisfi i requisiti espressi.

# Capitolo 2

## Analisi

Il progetto si compone di quattro parti:

- L’ **ontologia del dominio relativo al linguaggio c,** la quale cerca di descrivere in modo esaustivo il linguaggio C e quindi la struttura dei sorgenti C.
- Il **componente annotatore,** incaricato di trasformare il codice sorgente in ingresso in una rappresentazione semantica dello stesso.
- L’**ontologia del dominio relativo al materiale didattico,** la quale cerca di modellare in modo completo ogni aspetto relativo ai materiali didattici.
- Il **componente di raccomandazione,** incaricato di fornire materiale didattico in maniera coerente ai requisiti espressi e di suggerire materiale i cui argomenti sono strettamente correlati a quelli di interesse dell’utente.

## Contology

L’ ontologia Contology è stata realizzata dallo studente Diego Siboni anch’essa nell’ambito del corso Semantic Web ma nell’anno 2018-2019. Questa ontologia, strettamente correlata al linguaggio C, è stata realizzata con lo scopo di sperimentare un possibile approccio di annotazione semantica del codice sorgente C in maniera automatizzata. I principali concetti catturati dall’ontologia sono:

- Direttive.
- Funzioni.
- Espressioni.
- Tipi di dato.
- Variabilili.
- Modificatori.
- Dichiarazioni.

Inoltre, grazie a proprietà ad-hoc questi concetti vengono messi in relazione tra loro per la creazione di grafi RDF contenenti tutte le informazioni estrapolabili dal sorgente stesso. Per fare ciò la parte di automatizzazione è stata demandata ad un parser che preso in input un sorgente C fornisce in output un file .rdf contenente tutte le triple generate.

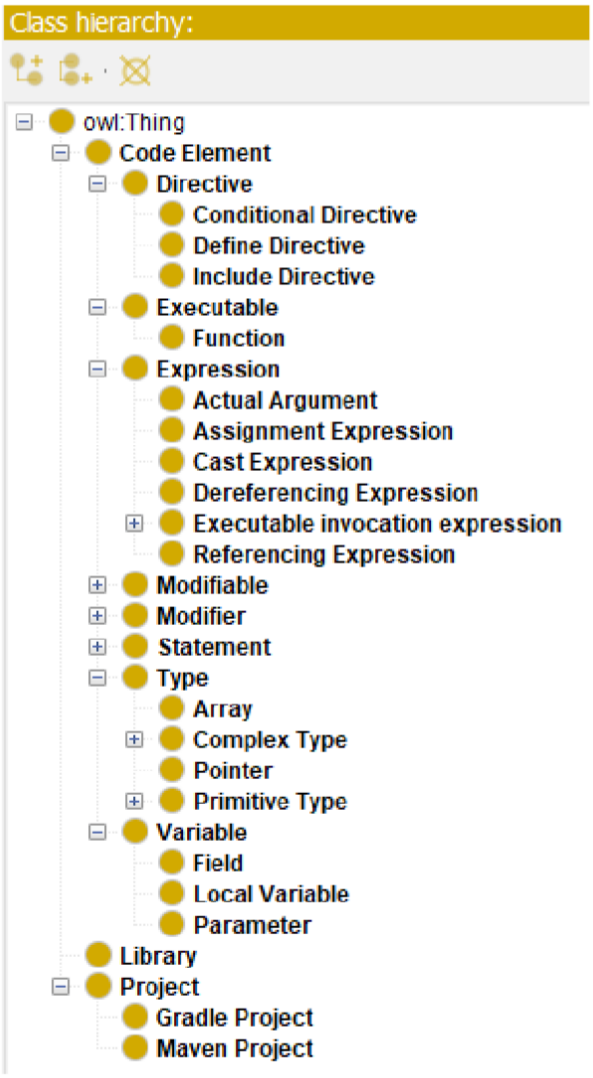


*Fig.1 – Struttura dell'ontologia C.*

## C-EducationalOntology

L’ontologia C-Educational è stata realizzata dallo studente Lorenzo Bacchiani nell’ambito del corso di Web Semantico nell’anno accademico 2018/2019. Lo scopo di questa ontologia è modellare nel dettaglio il dominio relativo ai materiali didattici relativi ad un ambito specifico come il linguaggio C.

Come si può veder in Fig. 2, i concetti principali sono:

- *Argument,* definisce i possibili temi dei materiali forniti. Questi topic sono presi dall’ontologia Contology.
- *DifficultyLevel*, modella il concetto di difficoltà del materiale fornito. Tale classe potrà assumere tre valori distingi: Facile, Medio, Difficile
- *EducationalLevel* viene utilizzato per definire il livello di educazione (Superiore o Universitario) a cui un determinato materiale di studio è adatto.
- *LearningObject* rappresenta il materiale vero e proprio.
- *LearningObjectType* modella il concetto di tipo di materiale, ad esempio un materiale didattico può essere scolastico o video o del codice.
- *Student* è una classe che modella il concetto di studente a cui verrà fornito il materiale sulla base dei requisiti espressi.
- *Source* definisce la fonte dei materiali forniti. In particolare, in questa ontologia le fonti dei vari materiali sono rappresentati come link al materiale preso da internet.

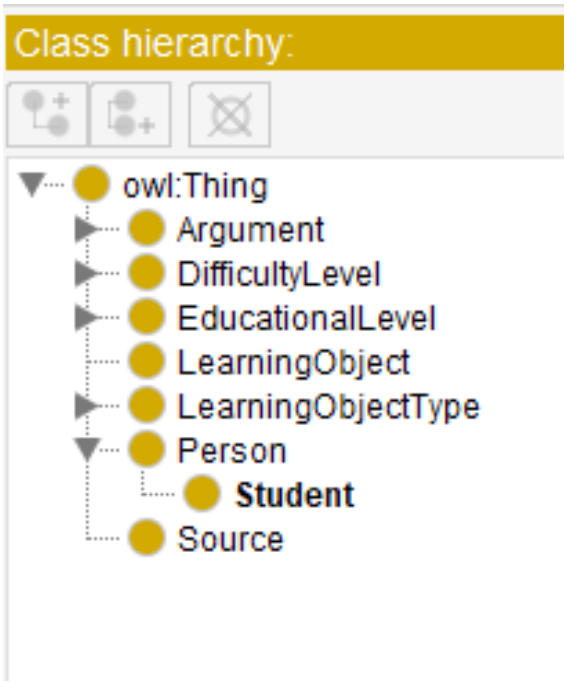

Fig. 2 – Concetti principali C-Educational Ontology.

Le principali proprietà che legano questi concetti sono le seguenti:

- *hasTopic*, lega un *LearningObject* all' *Argument* di cui tratta.
- *adaptTo,* collega un *LearningObject* all' *EducationalLevel* a cui è più adatto.
- *hasDifficulty,* è facilmente intuibile come questa proprietà colleghi un *LearningObject* al relativo livello di difficoltà.
- *hasSource,* connette un *LearningObject* alla relativa fonte.
- hasType, collega un LearningObject al tipo di materiale a cui appartiene (es. materiale scolastico).
- *hasInterest,* tale proprietà collega uno *Student* agli *Argument* che più le/gli interessano.
- *hasEducationalLevel,* collega uno *Student* al grado di educazione che possiede.
- *handleMaterialDifficultyLevel,* collega uno *Student* al livello di difficoltà del materiale che è in grado di gestire.
- *prefers,* collega uno *Student* al tipo di materiale più adatto a lei/lui.

Tutte queste proprietà potranno avere la contro parte passiva ad esempio *hasInterest* è la controparte attiva di *isInterestOf.* Inoltre per creare relazioni tra concetti correlati è stata usata la proprietà *related* presa da Simple Knowledge Organisation System (SKOS) che è una famiglia di linguaggi formali creata per rappresentare glossari, classificazioni ecc, basata su RDF e RDFS. La proprietà usata, *skos:related*, abilita la rappresentazione associativa non gerarchica di collegamenti come ad esempio le associazioni tra argomenti strettamente correlati come i vari for loop, while loop e do while loop.

## Annotatore semantico

Questa componente trasformerà sorgenti C in conoscenza su questi sorgenti. Esso dovrà quindi:

- Ricevere in ingresso un file di testo contenente un programma C.
- Restituire in uscita un *documento RDF* che descriva almeno tutte le righe del codice sorgente presentato.

- Utilizzare l’ontologia sviluppata per l’annotazione semantica.
- Restituire un errore in caso il codice sorgente fornito non sia corretto.

## Sistema di raccomandazione

Questo componente si occuperà di fornire il materiale materiale adatto allo studente sulla base di:

- Preferenze espresse sugli argomenti o argomenti presenti all’interno del file .c dato in pasto all’annotatore semantico.
- Preferenze espresse sul tipo di materiale voluto (materiale scolastico, esempi di codice video ecc.).
- Tipo di difficoltà del materiale desiderata.
- Livello di educazione espressa dallo studente.

Oltre al materiale direttamente collegato con gli argomenti di interesse il sistema di raccomandazione fornirà anche materiale i cui argomenti risultano strettamente correlati con quelli di interesse sempre rispettando i vincoli espressi dall’utente.

## Tecnologie

Per lo sviluppo del progetto si sono individuate delle tecnologie di riferimento da utilizzare:

- **RDF, RDFS, OWL,** come linguaggi per lo sviluppo dell’ontologia C-Educational.
- **JVM,** come base per lo sviluppo del sistema di raccomandazione;
- **SPARQL,** per le interrogazioni dell’ontologia.

# Capitolo 3

## Implementazione

Nella fase implementativa sono state fatte ulteriori scelte tecnologiche:

- **Java 11,** come versione di minima per la JVM per sfruttare alcune delle più recenti caratteristiche.
- **Kotlin,** come linguaggio di programmazione, per sperimentare una nuova tecnologia e un linguaggio moderno mantenendo comunque la retrocompatibilità con Java.
- **Eclipse CDT,** come parser di codice sorgente C siccome è utilizzabile in linguaggi JVM senza troppi problemi,
- **Apache Jena,** come gestore della rappresentazione e persistenza RDF e come motore per le interrogazioni SPARQL.

## C-Educational Ontology

Lo sviluppo dell'ontologia `e stato supportato dall'editor Protegè. Per modellazione dei concetti del linguaggio C si è preso spunto direttamente dalla Contology riportando la stessa suddivisione in classi. Al contrario per modellazione dei concetti relativi al materiale didattico si è partiti da una ontologia esistente mostrata in Fig. 3 e modificandola a dovere.

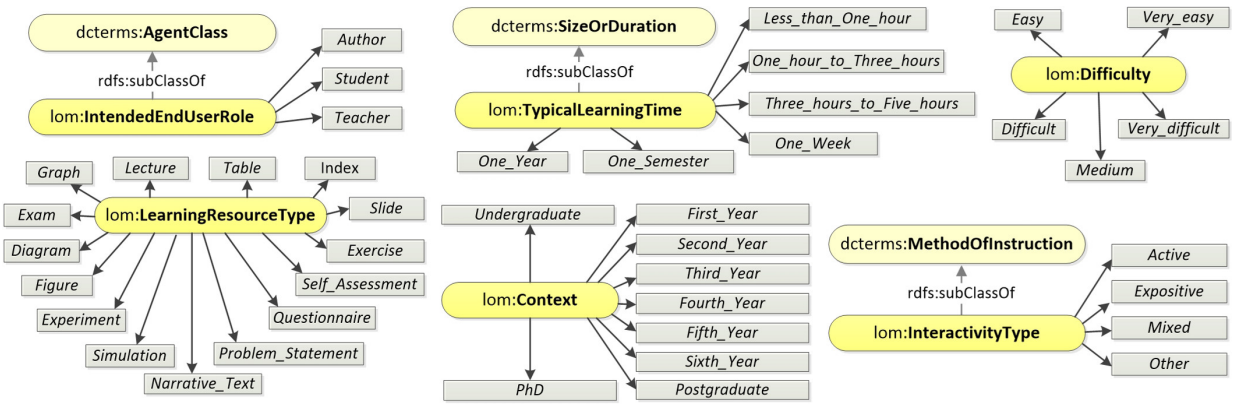


*Fig. 3 – Ontologia di partenza*

In particolare sono stati eliminati i concetti non utili alla modellazione del dominio del problema come:

- InteractiveType
- MethodOfInstructions
- TypicalLearningTime
- IntendedEndUserRole

Mentre sono stati aggiunti concetti come:

- Source

## Sistema di raccomandazione

Sia la componente che riguarda l'annotatore semantico sia il sistema di raccomandazione utilizzano il build system **gradle** che oltre ad occuparsi di gestire le dipendenze del progetto, permette anche la sua esecuzione da riga di comando. Il sistema può lavorare in due modalità, la prima prende in input un programma .c, richiama l'annotatore semantico che, oltre a produrre in output il file .rdf contenete le annotazioni, fornisce in input al sistema di raccomandazione tutti gli argomenti trattati nel file in input. A partire da questi argomenti, il sistema in prima battuta interroga, utilizzando il framework **Apache Jena**, il modello relativo alla C-Educational Ontology e stampando nello stdoutput argomenti e link ai materiali ad essi collegati. Un esempio di questo tipo di query è possibile trovarlo in Fig. 4.

SPARQL query:

```
PREFIX  edu: <http://www.semanticweb.org/lori/ontologies/2020/3/C-Educational-Ontology#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX skos:<http://www.w3.org/2004/02/skos/core#>
SELECT ?arg ?source
WHERE {
?material edu:hasSource ?source.
?material edu:hasTopic ?arg.
VALUES ?arg  { edu:ARGUMENT}.
}
```

*Fig. 4 – Esempio di query.*

In seguito verrà poi creata una nuova query per individuare il materiale i cui argomenti sono strettamente correlati a quelli individuati nella fase precedente e, come sopra, anch'essi verranno resi visibili nello stdoutput. Un esempio di questo tipo di query è possibile trovarlo in Fig. 5.

SPARQL query:

```
PREFIX  edu: <http://www.semanticweb.org/lori/ontologies/2020/3/C-Educational-Ontology#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX skos:<http://www.w3.org/2004/02/skos/core#>
SELECT ?arg ?source
WHERE {
    ?arg1 skos:related ?arg.
    ?relatedObj edu:hasTopic ?arg.
    ?relatedObj edu:hasSource ?source.
    VALUES ?arg1  { edu:TOPIC}.
}
```

*Fig. 5 – Esempio di query.*

In entrambe le query *ARGUMENT* è una variabile che a run time assumerà valori corrispondenti agli argomenti presenti nel file .c preso in input.

La seconda modalità di lavoro, consente di generare una quantità definita dall'utente di studenti con le relative preferenze su argomenti, tipo di materiale, livello di difficoltà e livello di scolarizzazione assegnati in maniera casuale. In seguito poi per ognuno di essi verrà poi effettuata una interrogazione per recuperare il materiale corrispondente ai vincoli espressi.

## Modifiche principali

Grazie alla struttura del parser, non è stato necessario modificare la prima fase (parsing automatico dalla libreria) ma la via migliore per l'arricchimento è stata quella di inserire ulteriore codice principalmente nella fase di mapping, per generare un collegamento diretto tra le due ontologie. Durante il mapping tra l'AST e le triple dell'ontologia cOntology sono state inserite ulteriori istruzioni per andare ad identificare gli argomenti che verranno poi utilizzati dal sistema di raccomandazione per fornire del materiale utile. Ciò è stato possibile creando nella classe adibita a questo mapping un campo denominato topics di tipo *MutableSet<String>* e richiamando in posizioni strategiche la funzione per aggiungere elementi a questo campo, dove tali elementi sono gli argomenti presenti all'interno del file .c dato in pasto al parser.

# Capitolo 4

## Prerequisiti

- Avere installato sul proprio sistema operativo Java 11 o successiva versione.
- Avere, almeno la prima volta, una connessione a internet funzionante (per scaricare il sorgente dell'ontologia).

## Avvio

Per utilizzare lo strumento:

- Posizionarsi nella "root folder" del progetto
- Aprire un terminale
- Eseguire il comando: *gradlew build*
- Per ottenere il materiale a partire da un file .c eseguire: *gradlew :run –args="A PathToCFile PathToRDFFile"*. Mentre per generare gli studenti e ottenere il materiale per ciascuno di loro eseguire: *gradlew :run –args="G Nstudenti"*, dove Nstudenti è il numero degli studenti che si intende generare.

**NOTA 1:** Se su Windows 10 viene utilizzato PowerShell il comando cambia in: .*gradlew run –args='A PathToCFile PathToRDFFile'*

**NOTA 2:** I due percorsi passati come argomenti in input devono essere senza spazi, oppure se contenenti spazi devono essere circondati da virgolette con un backslash davanti (\"PathWithSpaces\").

**NOTA 3:** L'utente che lancia il comando da terminale deve avere i permessi di lettura e scrittura rispettivamente per il percorso di input e quello di output forniti, altrimenti si avrà un errore; di solito la "home" dell'utente `e un buon inizio per effettuare i primi esperimenti.

## Esempio

Di seguito verrà mostrato un esempio del funzionamento di entrambe le modalità eseguite su sistema operativo *Windows 10* con *java 11* installato.Per la prima modalità di esecuzione, il codice che verrà dato in pasto al sistema è memorizzato nel seguente file ForLoop.c

ForLoop.c

```c
# include <stdio.h>

int main()
{

    const int maxInput = 5;
    int i;
    double number, average, sum=0.0;

    for(i=1; i<=maxInput; ++i)
    {
        printf("%d. Enter a number: ", i);
        scanf("%lf",&number);
        sum += number;
    }

    return 0;
}

```

*Fig. 6 – ForLoop.c*

Il file verrà “parsato” e verranno restituiti i materiali relativi ai concetti contenuti eseguendo i passi mostrati nella Fig.7.

```
Lori@DESKTOP-1L6R9HU MINGW64 ~/Desktop/web semantico/assignment2/ConsegnaBacchiani/recommender-C-system
$ gradle :run --args="\"A\" \"../CSOurceSamples/ForLoop.c\" \"annotation.rdf\""
> Task :compileKotlin UP-TO-DATE
> Task :compileJava NO-SOURCE
> Task :processResources UP-TO-DATE
> Task :classes UP-TO-DATE

> Task :run
Starting annotation mode...
Creating C AST...
C code AST created, checking for problems...
Code checked.
Created resulting annotation file at annotation.rdf
[IncludeDirective, Function, Const, ForStatement, ReferencingExpression, ReturnStatement]
--------Material recommended--------
Material for --> IncludeDirective
 - material at https://stackoverflow.com/questions/464560/how-to-use-include-directive-correctly
Material for --> Function
 - material at http://www.programmazione.info/lucidi_16_le_funzioni_parte_i.709.html
 - material at http://www.programmazione.info/lucidi_17_le_funzioni_parte_ii.711.html
 - material at http://www.programmazione.info/lucidi_18_le_funzioni_parte_iii.712.html
 - material at http://www.programmazione.info/lucidi_19_le_funzioni_parte_iv.716.html
Material for --> Const
 - material at https://www.geeksforgeeks.org/const-qualifier-in-c/
Material for --> ForStatement
 - material at https://www.youtube.com/watch?v=FPjLbPu5BsQ
Material for --> ReferencingExpression
 - material at https://www.html.it/pag/15511/dichiarazione-di-variabili-puntatore/
Material for --> ReturnStatement
 - material at https://docs.microsoft.com/it-it/cpp/c-language/return-statement-c?view=vs-2019
--------Related material--------
Material for --> DefineDirective
 - material at https://docs.microsoft.com/it-it/cpp/preprocessor/hash-define-directive-c-cpp?view=vs-2019
Material for --> ConditionalDirective
 - material at https://www.cs.auckland.ac.nz/references/unix/digital/AQTLTBTE/DOCU_078.HTM
Material for --> Volatile
 - material at https://en.wikipedia.org/wiki/Volatile_(computer_programming)
Material for --> GotoStatement
 - material at https://www.programiz.com/c-programming/c-goto-statement
Material for --> DoStatement
 - material at https://codeforwin.org/2015/06/for-do-while-loop-programming-exercises.html
Material for --> ContinueStatement
 - material at https://docs.microsoft.com/it-it/cpp/c-language/continue-statement-c?view=vs-2019
Material for --> BreakStatement
 - material at http://www.science.unitn.it/~fiorella/guidac/guidac016.html
Material for --> Pointer
 - material at http://www.programmazione.info/lucidi_15_i_puntatori_iii_parte.708.html
 - material at http://www.programmazione.info/lucidi_13_i_puntatori_ii_parte.700.html
 - material at http://www.programmazione.info/lucidi_12_i_puntatori_i_parte.699.html
Material for --> DereferencingExpression
 - material at https://www.html.it/pag/15511/dichiarazione-di-variabili-puntatore/
 - material at https://en.wikipedia.org/wiki/Dereference_operator
```

*Fig. 7 – Output della modalità di esecuzione comprendente il parser.*

Mentre per la seconda modalità di esecuzione è necessario eseguire i passi illustrati in Fig.8

```
Lori@DESKTOP-1L6R9HU MINGW64 ~/Desktop/web semantico/assignment2/ConsegnaBacchiani/recommender-C-system
$ gradle :run --args="\"G\" \"2\""
> Task :compileKotlin UP-TO-DATE
> Task :compileJava NO-SOURCE
> Task :processResources UP-TO-DATE
> Task :classes UP-TO-DATE

> Task :run
Generating students...
-----------Material for AndreaTagliati-----------
Query parameter:
- Topics: [GotoStatement, WhileStatement, Const, AutoModifier]
- Educational level: University
- Difficulty level handled: [Easy, Hard]
- Preferred types: []
Recommended material:
Material for --> AutoModifier
 - material at https://riptutorial.com/c/example/12392/auto
Material for --> Const
 - material at https://www.geeksforgeeks.org/const-qualifier-in-c/
You may be interested in:
Material for --> RegisterModifier
 - material at https://www.webmasterpoint.org/programmazione/programmazione/c/variabili-register.html
Material for --> DoStatement
 - material at https://codeforwin.org/2015/06/for-do-while-loop-programming-exercises.html
Material for --> ContinueStatement
 - material at https://docs.microsoft.com/it-it/cpp/c-language/continue-statement-c?view=vs-2019
Material for --> BreakStatement
 - material at http://www.science.unitn.it/~fiorella/guidac/guidac016.html
-----------Material for CarlaAbbruciati-----------
Query parameter:
- Topics: [Enum, ExternModifier, ReferencingExpression, Const, Pointer]
- Educational level: University
- Difficulty level handled: []
- Preferred types: []
Recommended material:
Material for --> Pointer
 - material at http://www.programmazione.info/lucidi_15_i_puntatori_iii_parte.708.html
 - material at http://www.programmazione.info/lucidi_13_i_puntatori_ii_parte.700.html
 - material at http://www.programmazione.info/lucidi_12_i_puntatori_i_parte.699.html
Material for --> Const
 - material at https://www.geeksforgeeks.org/const-qualifier-in-c/
Material for --> ReferencingExpression
 - material at https://www.html.it/pag/15511/dichiarazione-di-variabili-puntatore/
Material for --> Enum
 - material at https://www.geeksforgeeks.org/enumeration-enum-c/
You may be interested in:
Material for --> Volatile
 - material at https://en.wikipedia.org/wiki/Volatile_(computer_programming)
Material for --> DereferencingExpression
 - material at https://www.html.it/pag/15511/dichiarazione-di-variabili-puntatore/
 - material at https://en.wikipedia.org/wiki/Dereference_operator
Material for --> StaticModifier
 - material at https://en.wikipedia.org/wiki/Static_(keyword)
Material for --> RegisterModifier
 - material at https://www.webmasterpoint.org/programmazione/programmazione/c/variabili-register.html
Material for --> AutoModifier
 - material at https://riptutorial.com/c/example/12392/auto
Material for --> Union
 - material at https://www.tutorialspoint.com/cprogramming/c_unions.htm
Material for --> Struct
 - material at https://www.programiz.com/c-programming/c-structure-examples
```

*Fig. 8 – Output della modalità di esecuzione che genera gli studenti.*

Come si può notare dalle Fig. 7 e Fig. 8 l'output viene fornito sotto forma di link a cui lo student può trovare del materiale didattico che rispetti i vincoli da lui/lei espressi.

# Capitolo 5

## Conclusioni

Il progetto ha dimostrato la realizzazione di un sistema di raccomandazione di materiale didattico basato su ontologia. Tale strumento in combinazione con l'annotatore semantico si dimostra particolarmente utile in quanto, in maniera totalmente automatizzata, è possibile fornire del materiale didattico allo studente a partire da un qualsiasi sorgente, o in alternativa, è possibile fornire allo studente il materiale che più si adatta alle sue preferenze e conoscenze.

## Lavori futuri

Il sistema realizzato pone le basi per possibili lavori futuri e miglioramento del progetto:

- **Arricchimento dell'ontologia.** Un elemento fondamentale per la qualità del sistema è l'ontologia ed il materiale ad esso collegato pertanto più materiale viene raccolto all'interno di essa più il sistema risulterà completo e adatto ad ogni tipo di vincolo espresso dallo studente.
- **Creazione interfaccia grafica.** Non essendo un requisito fondamentale per la realizzazione di un sistema del genere si è deciso di adottare come interfaccia la riga di comando perciò non è da escludere in futuro la creazione di un'interfaccia grafica più intuitiva e comoda.
- **Collegamento con un database,** da cui recuperare il materiale didattico ed utilizzarlo per mantenere salvati i profili degli utenti per velocizzare le ricerche dei materiali.