* **Progettazione e documentazione** (file di testo, word, html, o altro) prodotta con le tecniche/strumenti di design imparati nel corso di Ingegneria del Software, relativamente a
  + Requisiti del sistema (funzionalità ed altri requisiti non funzionali oltre ad eventuali vincoli di implementazione e di distribuzione)
  + Progettazione dell'architettura (“in grande”) con sviluppo agile e UML (casi d’uso e diagramma delle componenti/deployment)
    - Uso di **UML** (diagrammi delle classi e altri diagrammi) e **ER**, **Reti di Petri**, **Abstract State Machines** eventualmente per le funzionalità più critiche
  + Progettazione “in piccolo” di **algoritmi e strutture dati**
  + Piano di test e spiegazione delle decisioni prese durante l'implementazione
  + Manuale d’uso e di installazione
* **Implementazione**
  + Costrutti OO di Java: uso di interfacce e classi astratte, famiglie di costruttori, ereditarietà, overloading e overriding di metodi, membri static, final, applicazione dei design pattern, generics (tipi o metodi generici), ecc.
  + Algoritmi: algoritmi user-defined, uso di **JFC** (tra cui **List** e **HashTable**)
  + Librerie esterne: ad es. **log4j** per il controllo dell’accesso o altre (**jcurses**, **jexcelapi**, o di **Jakarta**)
  + Unit testing: sviluppare e documentare casi di test con **Junit**
  + Copertura: copertura del codice con **Emma** (o altri tool) e tale copertura deve essere documentata
  + Documentazione del codice: completamente documentato con JavaDoc (tool JAutoDoc)
  + I/O in formato XML: con **SAX o DOM** (o utilizzare un parser generator come AntLR o Javacc)
  + GUI: un’interfaccia grafica (differenziare la UI con le funzionalità dell'applicazione); preferite l'uso del pattern Model View Controller (MVC)
* **Distribuzione**
  + Al di fuori dell'IDE: installazione e funzionamento stand-alone con un installer o con uno script o con Java web start. Usate fatjar se volete mettere più jar in un unico Jar.
  + In linea di principio, non deve richiedere al docente l’installazione di SW ausiliario (DBMS o altro) per farlo funzionare. Se necessario, preferite l'uso di Java DB (http://developers.sun.com/javadb/ , solo 2MB!) e notificatelo al docente.
  + Il codice deve essere sviluppato (e consegnato al docente) in un (unico) progetto di Eclipse e deve includere librerie e risorse necessarie per compilare/eseguire i file e i casi di test.
    - I link alle risorse devono essere *relativi* in modo che spostando il progetto su vari PC esso continui a funzionare. Se utilizzate altri progetti ausiliari, anche questi vanno inclusi.