1) Testing di un programma
scrivere i casi di test per coprire: [22-03-02 1:5 circa]
-istruzioni (righe di codice)
-condizioni (atomi dentro le decisioni)
-decisioni
-branch
-mcdc (conviene farlo per le singole decisioni e poi integrarli)
-minimizzare i casi di test
-casi di test convertibili in junit (scrivere del codice per chiamare la funzione in quel modo)

2) Algoritmo model checking [22-05-11]
-in quali stati valgono delle proprietà date
-AG(p) > not EF(p) > not E(T U not p)
-EF(p) > E(T U p)
-ci sono {A,E} e {G,F,X,U,W}

3) Combinatorial Testing [22-05-13 1:00 circa]
-date variabili costruire la test suite pairwise usando algoritmo IPO
-costruire test suite compatta e allungata (numero di casi da usare dato dal testo)
-modificare in base a delle esclusioni date dal testo

4) Conformance Testing [22-05-18 spiega tutto, verso la fine fa tema d'esame con how-to]
data FSM dire
-se e' correttamente definita = se, per ogni stato e per ogni input, ogni stato usa tutti gli input e so dove va
-test sequences per stati e per transizioni = copro stati e poi transizioni (cerco di farla minima), considera che parto da s0
-esempi di errori (visti) = errore output e errore di transfer
-controllare se si riescono a scoprire gli errori = se faccio errori di output potrei non trovarli se per coprire gli stati non passo da quella strada, ma con la ts delle transizioni (tt) lo trovo. se faccio errori di transfer (sull'ultimo passo per esempio) non lo trovo ne con ts per stati ne per transizioni, ma solo con status.

5) JML e junit
-creazione contratti (jml)
-provare la correttezza (junit)

6) Logica temporale ASMETA
-modellare stati dati dal testo (asmeta)
-provare proprietà testuali con LTL o CTL (asmeta)
-trovare proprieta false con controesempio (asmeta)
-creare scenari (avalla)
-convertire scenari avalla in java