# Test report:

## #1: Triangle

Files used:

* TriangleTests.hs (contains test variables and functions)
* Triangle.hs (contains the actual functions)
* Week2.hs (contains global functions used in the 2nd week)

We’ve created tests for:

* **NoTriangles**, by creating input lists which contain:
  + negative numbers, like [-1,2,3] [-1,-1,-1] etc.
  + one number that was bigger than the smaller two together, like [10,30,50].
* **Equilateral**, by creating input lists which contain:
  + Three equal numbers, like [1,1,1] [2,2,2] etc.
* **Isosceles**, by creating input lists which contain:
  + Two equal numbers and one different, like [20,20,30] [20,30,20].
* **Rectangular**, by creating input lists which contain:
  + Three numbers that equal a^2+b^2=c^2, like [4,3,5] [96,28,100].
* **Other**, by creating input lists which contain:
  + A list of all combined numbers of one to 100 and do not have three or two same numbers, are not Pythagorean and are of valid lengths (two smallest sides together are bigger than the longest size)

Total time spent: about 2 hours.  
Note: more details can be found in the TriangleTests.hs itself.  
“allTriangleTests” can be used to test all tests.

## #2: Propositional

Files used:

* PropositionalTests.hs (contains test-‘Forms’ and test functions)
* Propositional.hs (contains the actual functions)
* Week2.hs (contains global functions used in the 2nd week)

We’ve created tests for:

* **Tautologies**, by creating 23 Forms (/propositional logic formula’s) and check if the functions returns true.
  + For example: testT1 = all tautology allTaut
* **Contradictions**, by creating 2 Forms and check if the functions returns true.
  + For example: testC1 = all contradiction [cont1, cont2]
* **Satisfiables**, by creating 3 Forms and check if the function returns true. We’ve also tested if the satisfiable Froms are not tautologies and are not contradictions. We used simple formula’s to ensure that these functions were not tautologies (or else this test would not have been valid)
  + For example: testS1 = all satisfiable allSatis
  + testS2 = not (any contradiction allSatis)
  + testS3 = not (any tautology allSatis)
* **Entail** forms, by creating 5 pairs of Forms and check if the functions returns true.
  + For example: testE1 = entails entails1a entails1b
* **Logical Equivalent** forms, by creating 4 pairs of Forms and check if the functions returns true.
  + For example: testEq1 = equiv equiv1a equiv1b

Total time spent: about 2 hours.  
Note: more details can be found in PropositionalTests.hs   
“allPropositionalTests” can be used to test all tests.

## #3: CNF

Files used:

* CNFTests.hs (contains test variables and test functions)
* CNF.hs (contains the actual functions)
* Week2.hs (contains global functions used in the 2nd week)

We’ve created tests for:

* **Distribution** function, the ones that were provided in the slides.
  + First case uses equivalence of (p ^ q) \_ r and (p \_ r) ^ (q \_ r).
  + Second case uses equivalence of p \_ (q ^ r) and (p \_ q) ^ (p \_ r).
* **Forms**, by executing the cnf function with all pre-used Forms and check if they are still logical equivalent.

Total time spent: about 4 to 5 hours.  
Note: more details can be found in CNFTests.hs   
“allCNFTests” can be used to test all tests.